

FUNDAMENTOS DA COMPUTAÇÃO



complexidade
de algoritmos

compilação

arrays

SQL

estruturas
de dados

ponteiros

algoritmos

fundamentos da programação

internet

web

linux

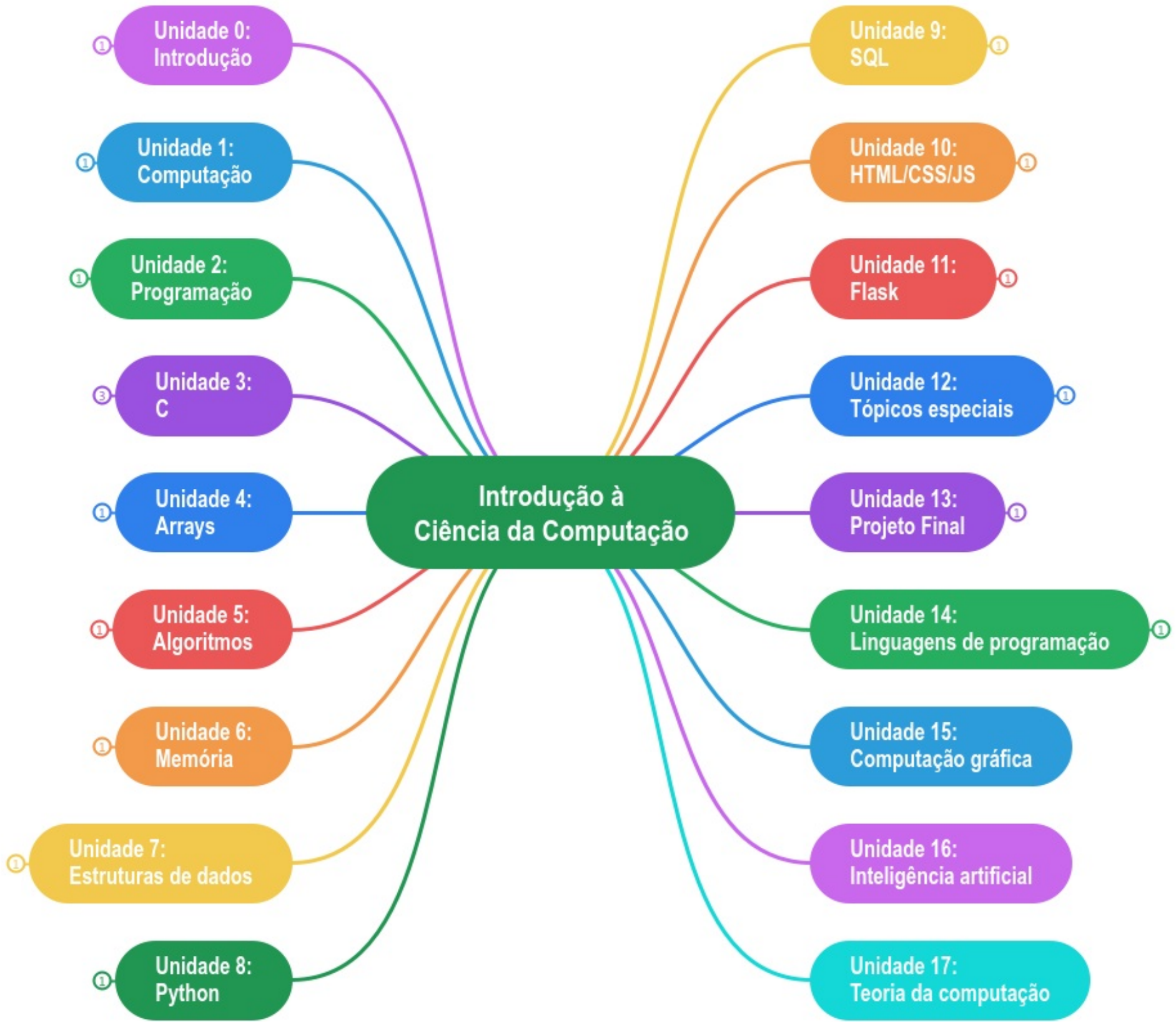
pensamento
computacional

memória

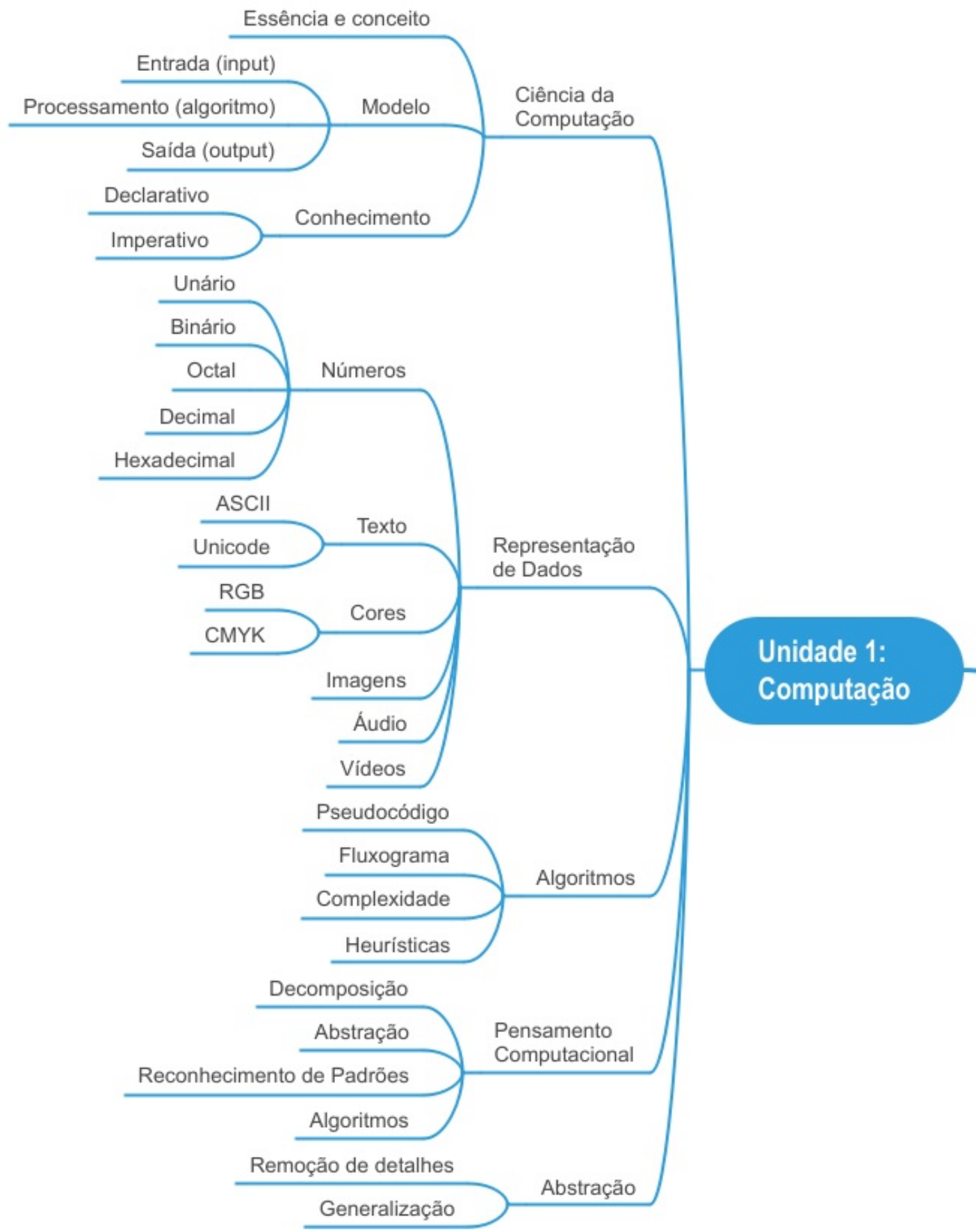
C

fundamentos da computação

Visão geral



Esta unidade



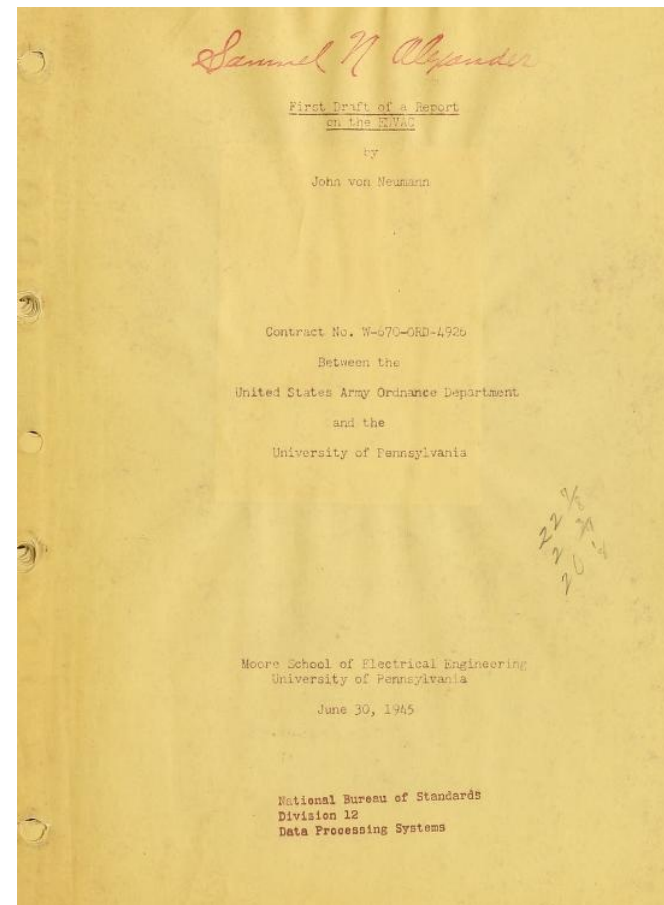
Representação de dados



Imagem: Los Alamos National Laboratory, na Wikipedia
(<https://en.wikipedia.org/wiki/File:JohnvonNeumann-LosAlamos.gif>)

John von Neumann ("von Neumann")

- **Binário**
- **Arquitetura de von Neumann**
- **Programação linear**
- **Computação científica**
- **Outras contribuições: matemática, física, engenharia, ...**

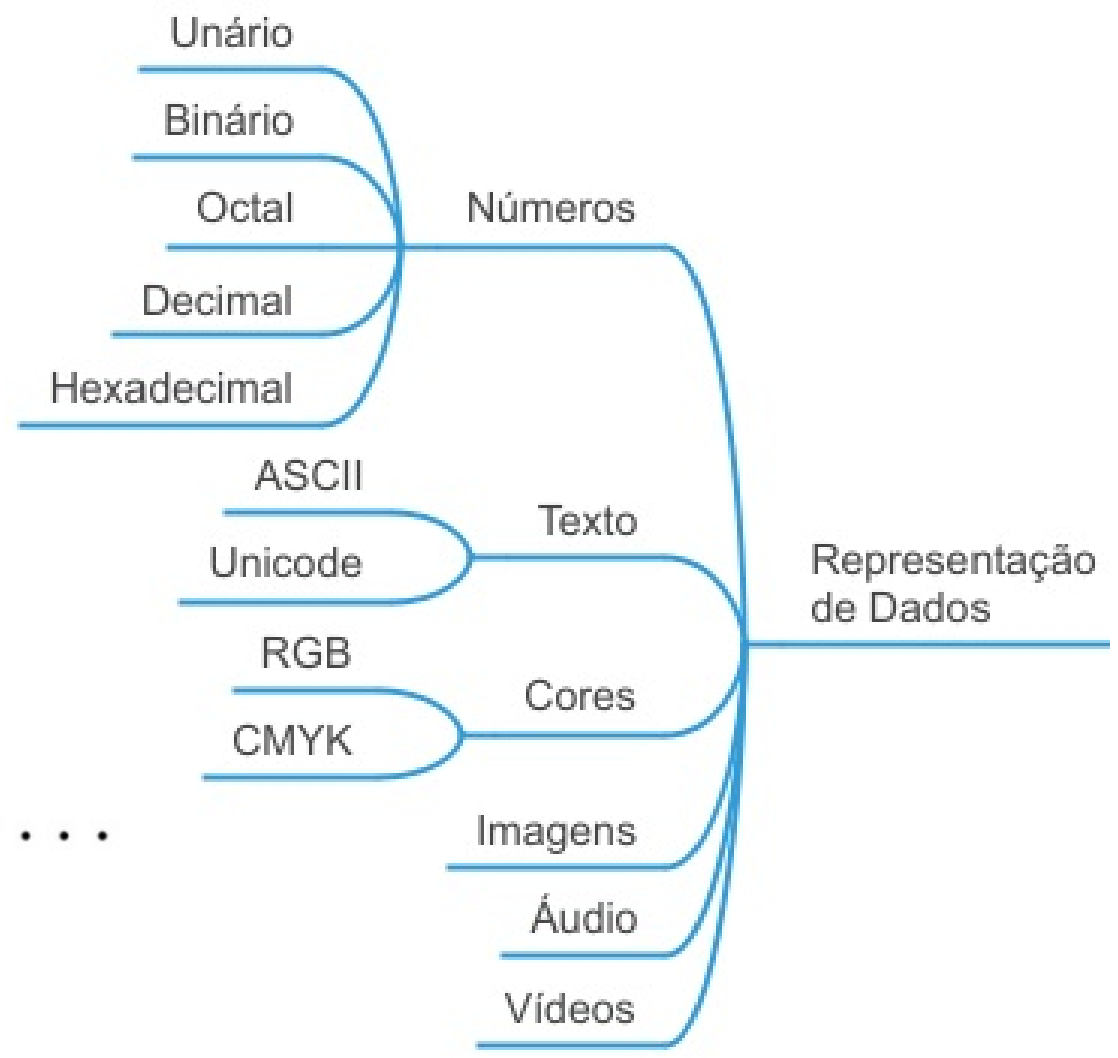
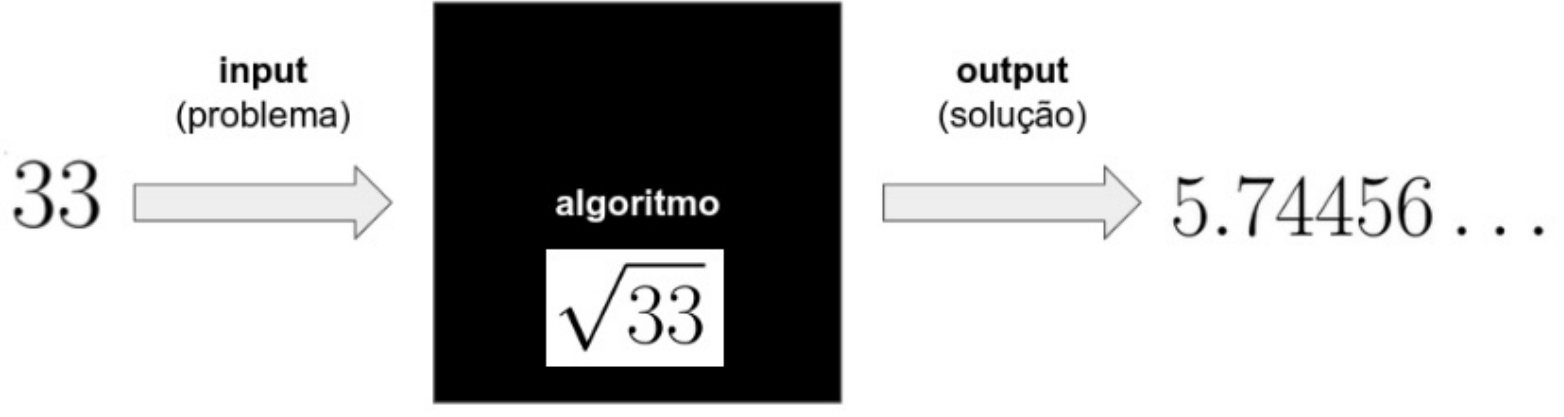
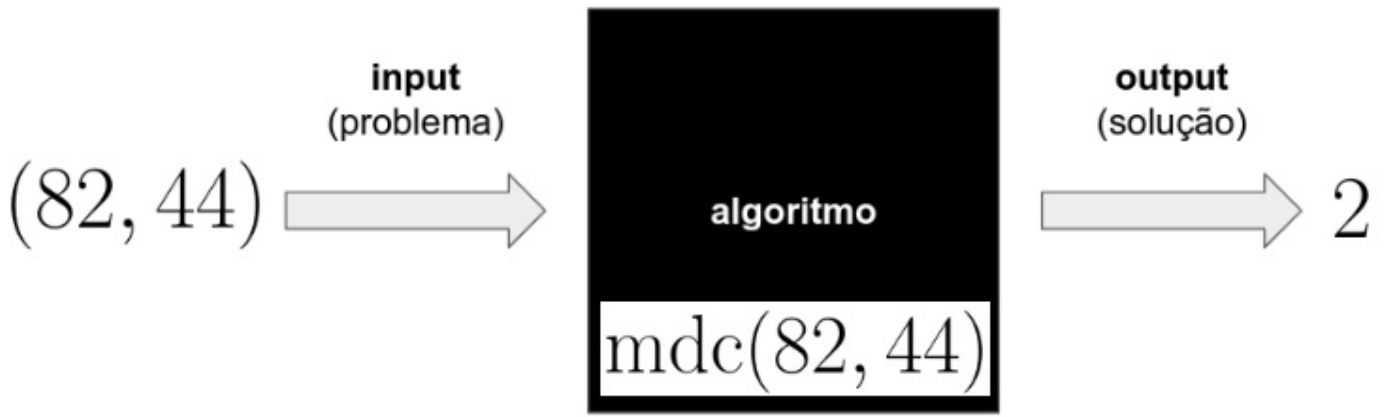


First Draft of a Report on the EDVAC
(30/06/1945)

Primeira descrição do projeto de um computador que usa o conceito de programa armazenado.

Imagem: Smithsonian Libraries and Archives, no Internet Archives
(<https://archive.org/details/firstdraftofrepo00vonn/>)

Por que a representação é tão importante?



Por que a representação é tão importante?

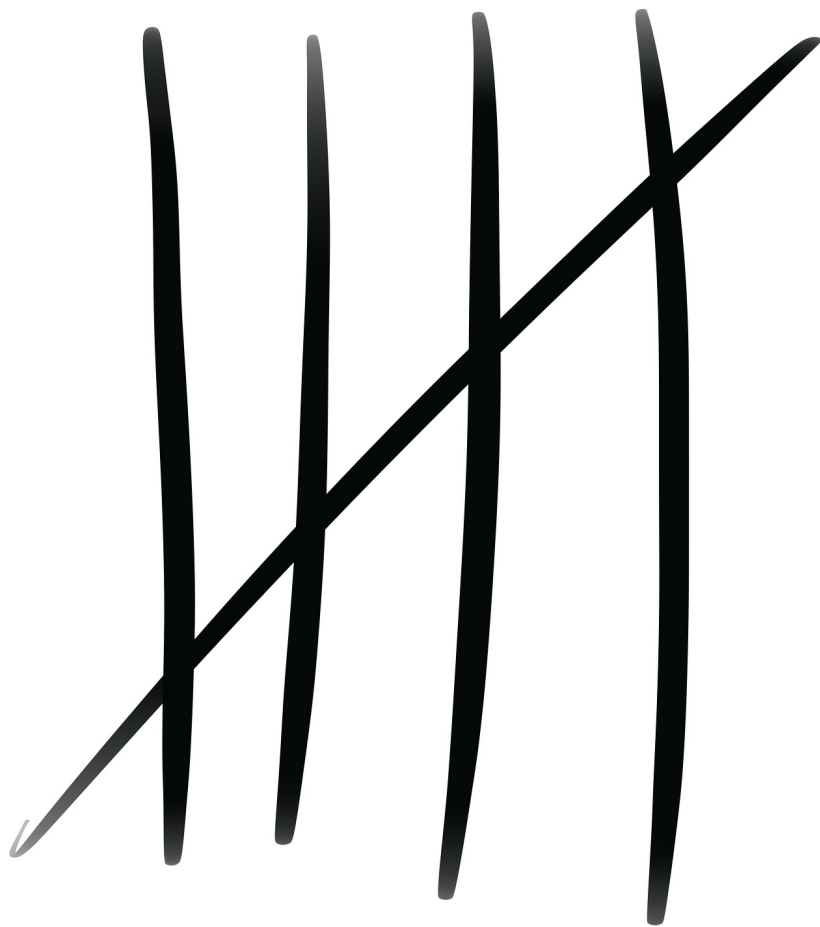


Imagem: StL20, no Pixabay

(<https://pixabay.com/illustrations/to-match-opinion-poll-count-six-6700969/>)



Imagem: ALLes, no Pixabay (<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)

Sistema unário: apenas um único algarismo, o "1"

Quantos números podemos representar com uma mão?

É possível fazer melhor? É possível representar mais com apenas uma mão?

Por que a representação é tão importante?



Imagem: ALes, no Pixabay
(<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)



Imagem: ALes, no Pixabay
(<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)

Sistema binário: dois algarismos, o "0" e o "1"

Quantos números podemos representar com uma mão?

Quantos números podemos representar com duas mãos?

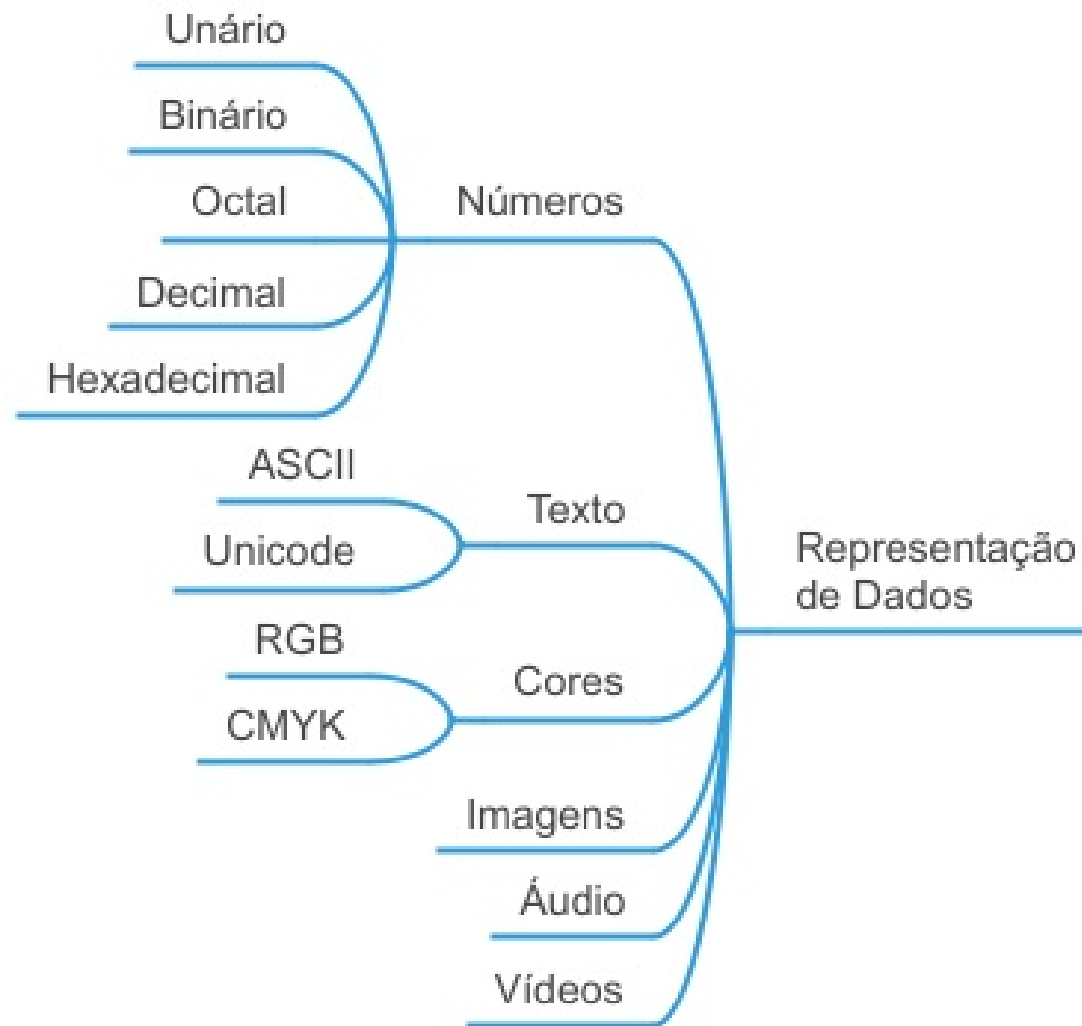
Por que a representação é tão importante?

Ao mudar a representação do problema:

- Podemos fazer mais com menos
- Podemos utilizar algoritmos mais adequados
- Temos mais flexibilidade da entrada e saída



Representação de dados: números



Noção intuitiva:

- Sistema unário
- Sistema decimal
- Sistema binário

Por que binário?

- Os 0s e 1s não existem, são abstrações!
- Transístores, campo magnético, sulcos, luz...

Formalização:

- Sistemas decimal, binário, octal e hexadecimal

Unidades de medida:

- Decimal
- Binário

Representação de números: noção intuitiva do sistema unário

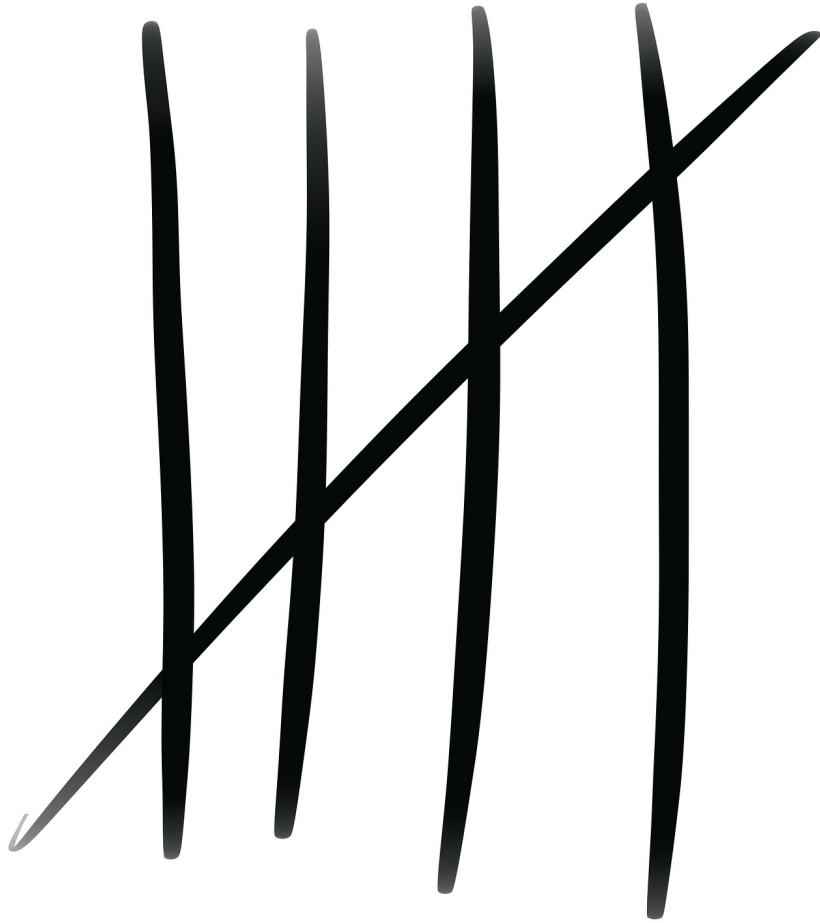


Imagem: StL20, no Pixabay
(<https://pixabay.com/illustrations/to-match-opinion-poll-count-six-6700969/>)



Imagem: ALes, no Pixabay (<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)

Sistema unário: apenas um único algarismo, o "1"
- Muito limitado para nossos propósitos

Representação de números: noção intuitiva do sistema binário



Imagem: ALes, no Pixabay

(<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)



Imagem: ALes, no Pixabay

(<https://pixabay.com/photos/hand-palm-fingers-one-two-three-4594071/>)

Sistema binário: dois algarismos, o "0" e o "1"
- Os computadores "falam" binário!

Representação de números: noção intuitiva do sistema binário

O sistema binário funciona como um hodômetro de um carro. Normalmente, com o sistema decimal (10 algarismos, do 0 ao 9), temos o seguinte:

Decimal: 0 0 0 0 0 0 0 0 0 0	Decimal: 0 0 0 0 0 0 0 0 0 5
Decimal: 0 0 0 0 0 0 0 0 0 1	Decimal: 0 0 0 0 0 0 0 0 0 6
Decimal: 0 0 0 0 0 0 0 0 0 2	Decimal: 0 0 0 0 0 0 0 0 0 7
Decimal: 0 0 0 0 0 0 0 0 0 3	Decimal: 0 0 0 0 0 0 0 0 0 8
Decimal: 0 0 0 0 0 0 0 0 0 4	Decimal: 0 0 0 0 0 0 0 0 0 9

Decimal: 0 0 0 0 0 0 0 0 1 0

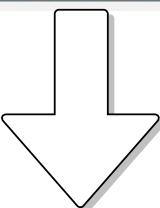
← Ao acabar os algarismos: "Zera e vai 1"

↓

Representação de números: noção intuitiva do sistema binário

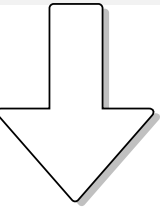
Decimal: 0000000019

Decimal: 0000000020



Decimal: 0000000029

Decimal: 0000000030



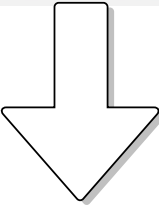
Decimal: 0000000039

Decimal: 0000000040

Ao acabar os algarismos:
"Zera e vai 1"

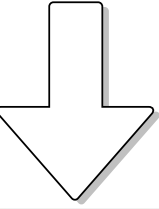
Decimal: 0000000099

Decimal: 0000000100



Decimal: 0000000999

Decimal: 0000001000



Decimal: 0000009999

Decimal: 0000010000

Representação de números: noção intuitiva do sistema binário

Com o sistema binário (2 algarismos, do 0 ao 1), temos o seguinte:

Ao acabar os algarismos:
"Zera e vai 1"



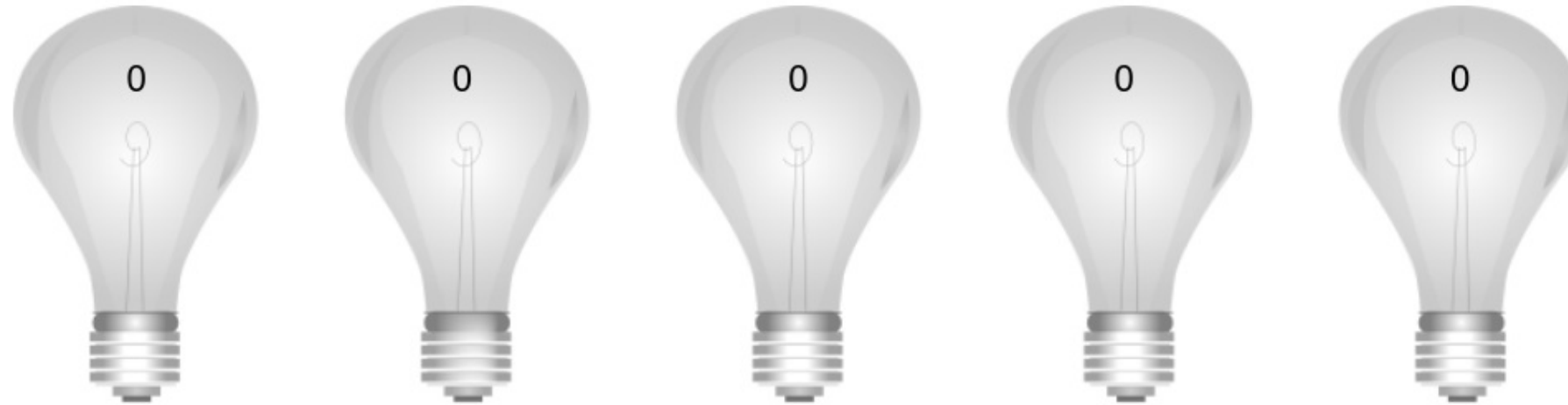
Cada "casa" para os algarismos é chamada de "bit" (binary digit).

Nesse hodômetro podemos armazenar números binários com 10 bits. Sendo assim, o maior número binário que pode ser armazenado é o



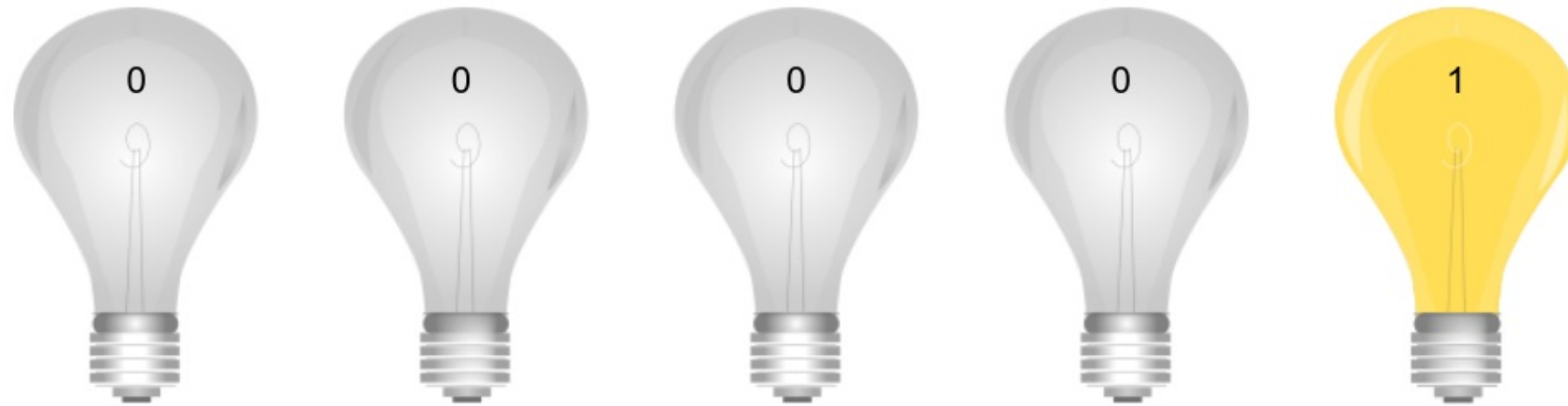
que equivale a 1023 em decimal.

Representação de números: noção intuitiva do sistema binário



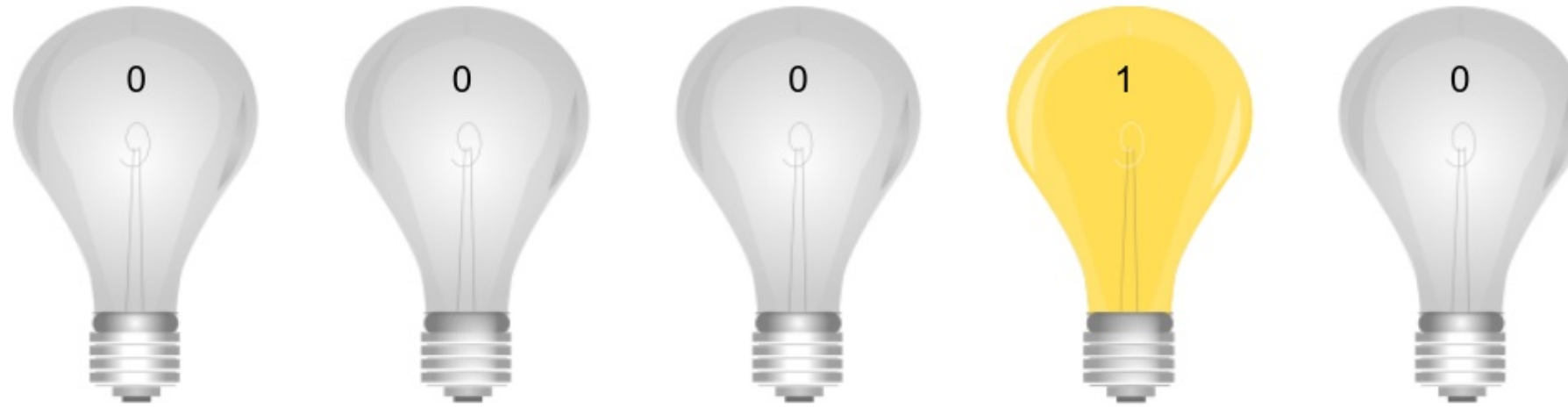
0

Representação de números: noção intuitiva do sistema binário



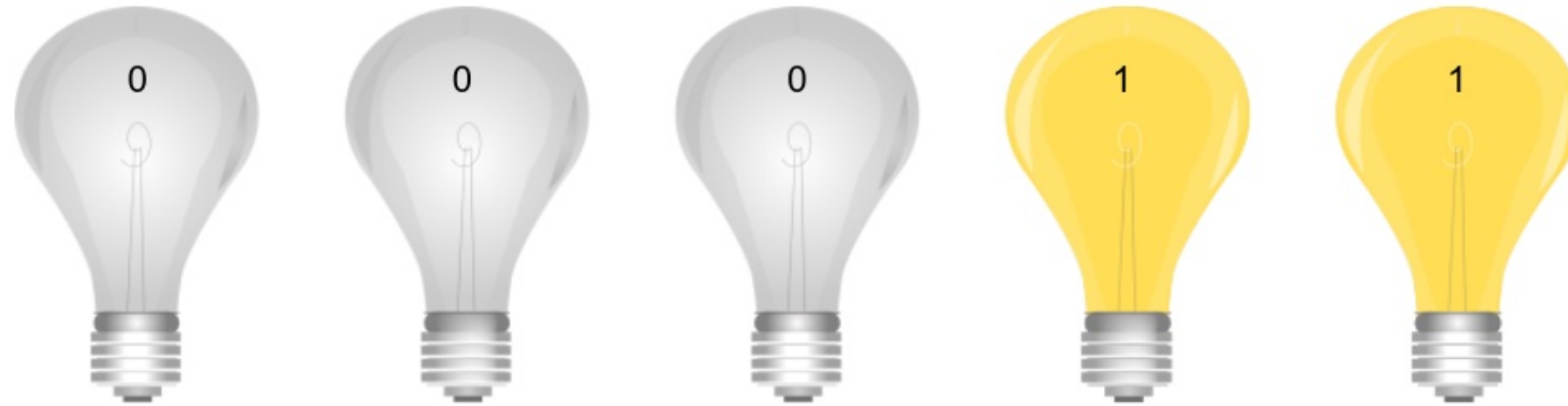
1

Representação de números: noção intuitiva do sistema binário



2

Representação de números: noção intuitiva do sistema binário



3

Representação de números: noção intuitiva do sistema binário



4

Representação de números: noção intuitiva do sistema binário



5

Representação de números: noção intuitiva do sistema binário



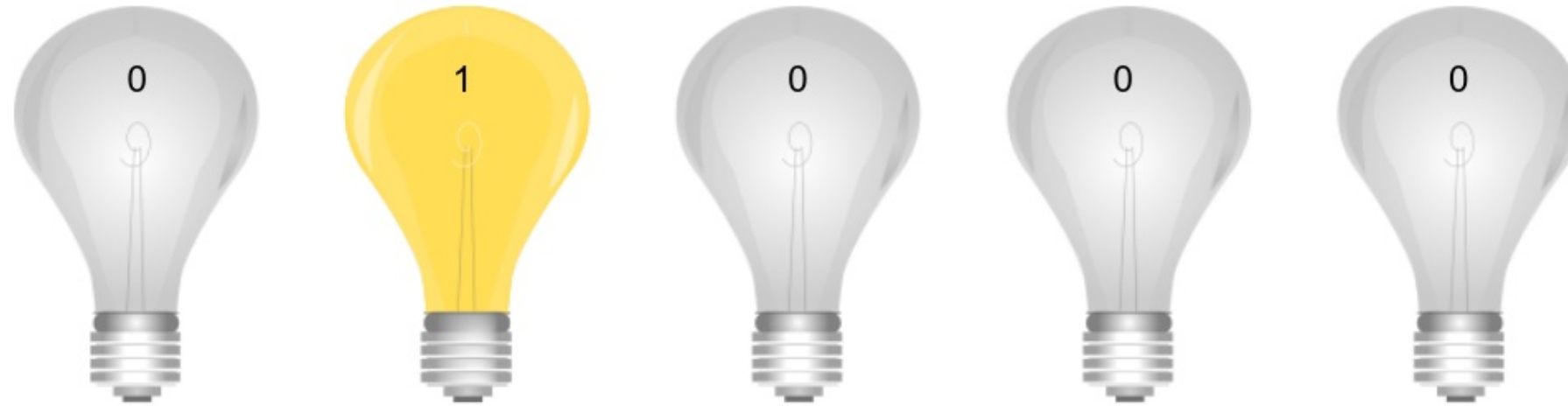
6

Representação de números: noção intuitiva do sistema binário



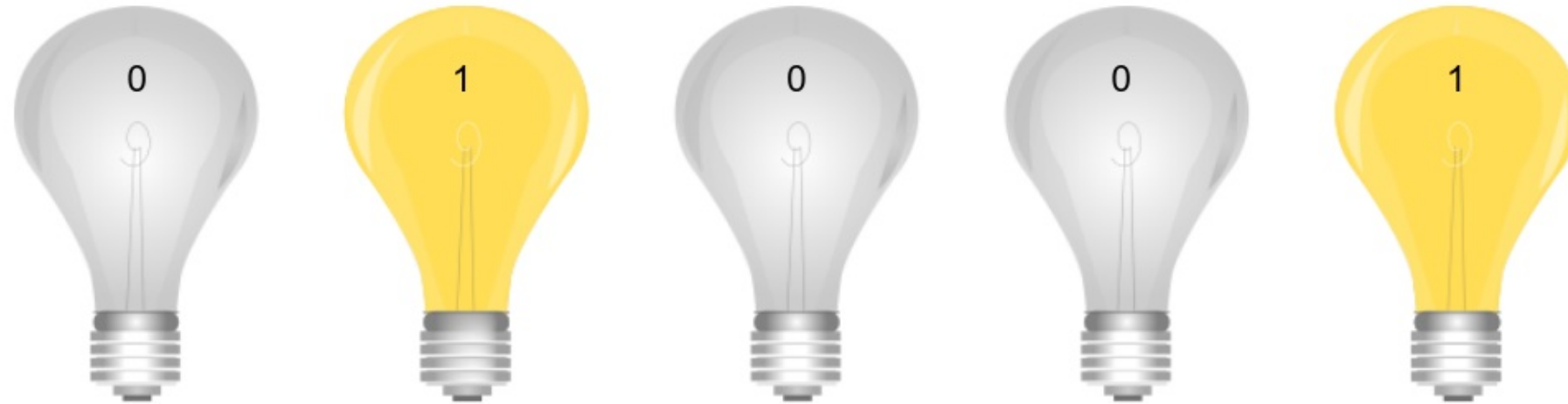
7

Representação de números: noção intuitiva do sistema binário



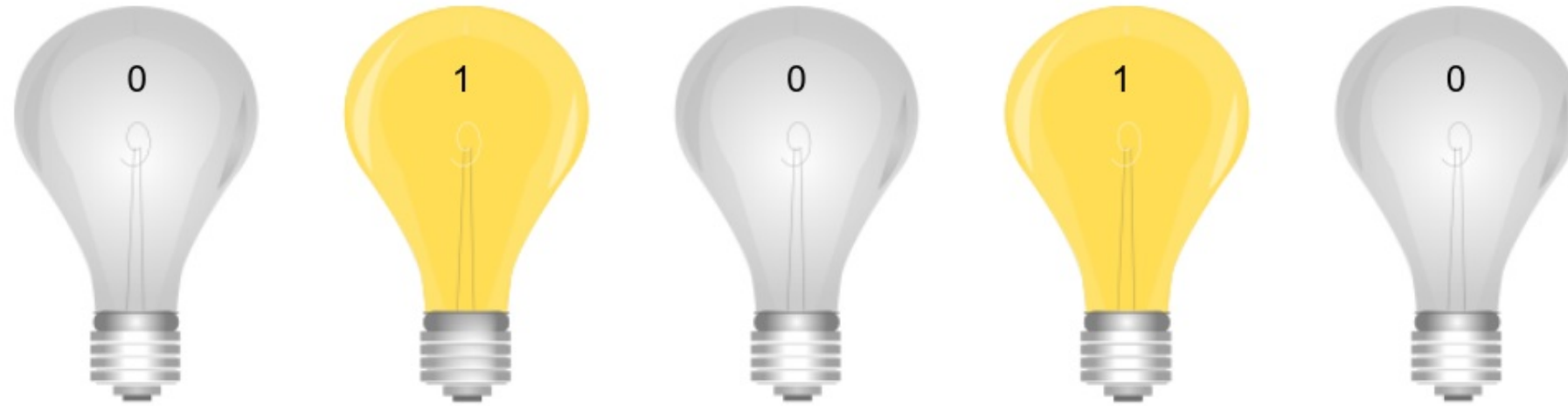
8

Representação de números: noção intuitiva do sistema binário



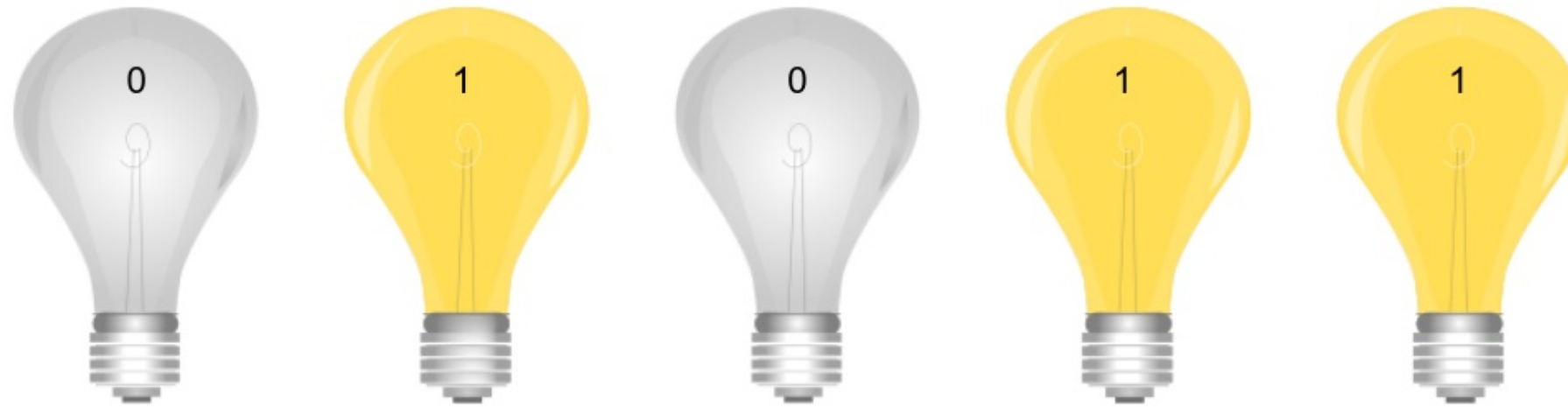
9

Representação de números: noção intuitiva do sistema binário



10

Representação de números: noção intuitiva do sistema binário



11

Representação de números: noção intuitiva do sistema binário



12

Representação de números: noção intuitiva do sistema binário



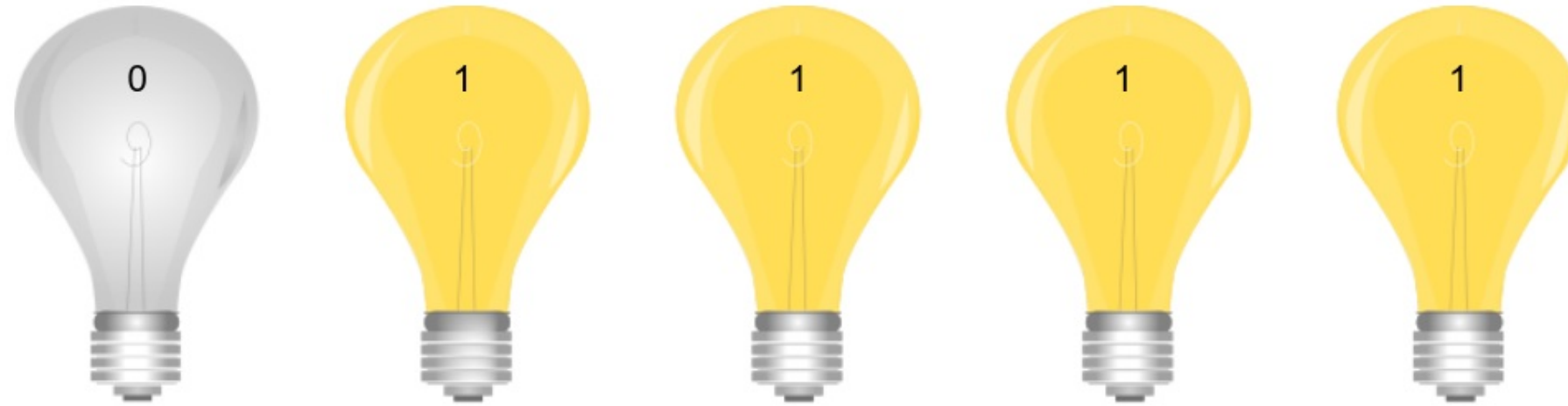
13

Representação de números: noção intuitiva do sistema binário



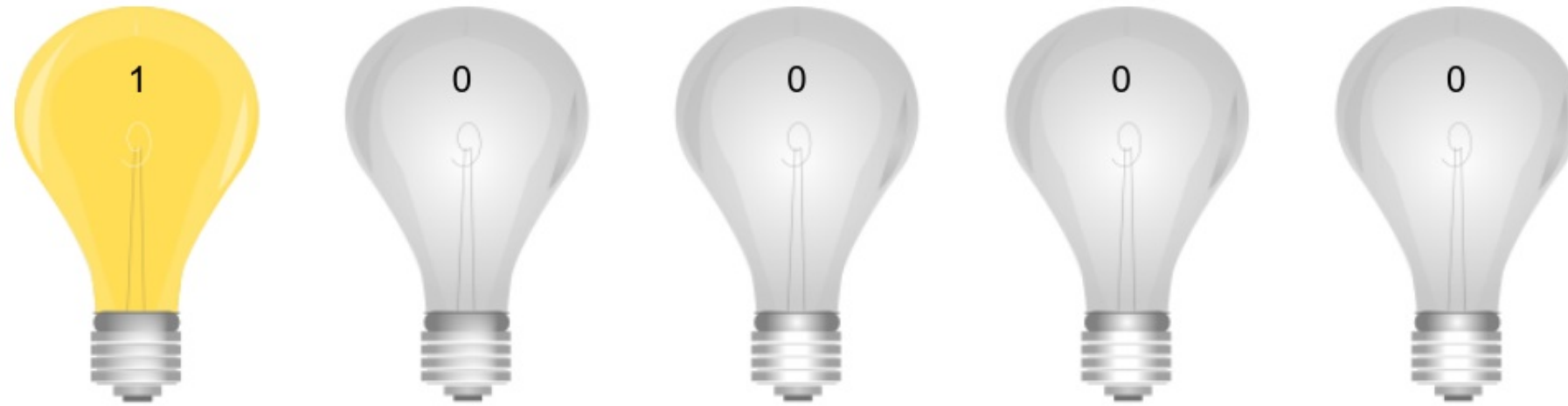
14

Representação de números: noção intuitiva do sistema binário



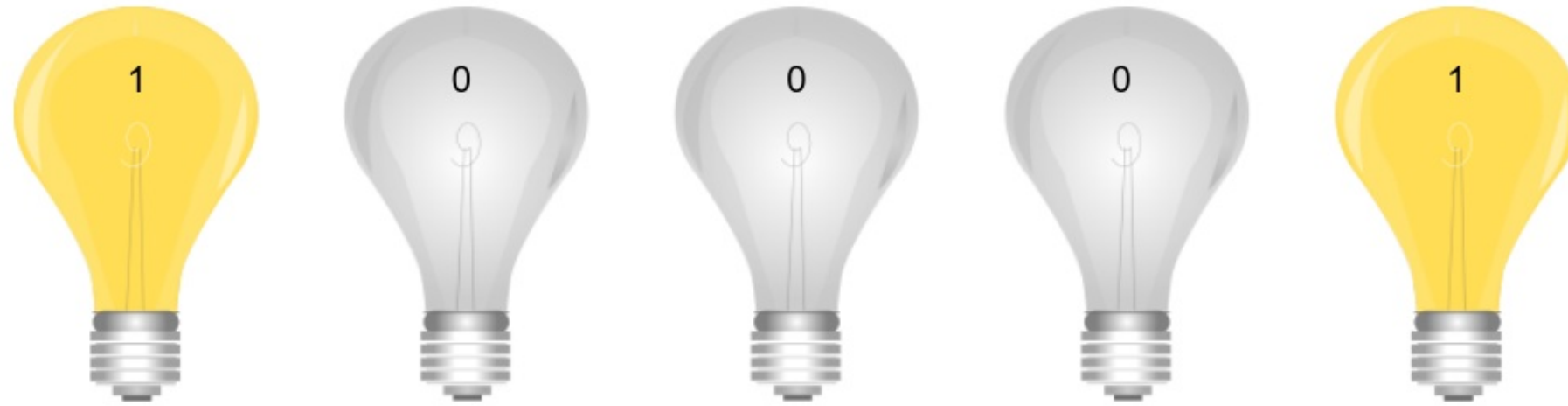
15

Representação de números: noção intuitiva do sistema binário



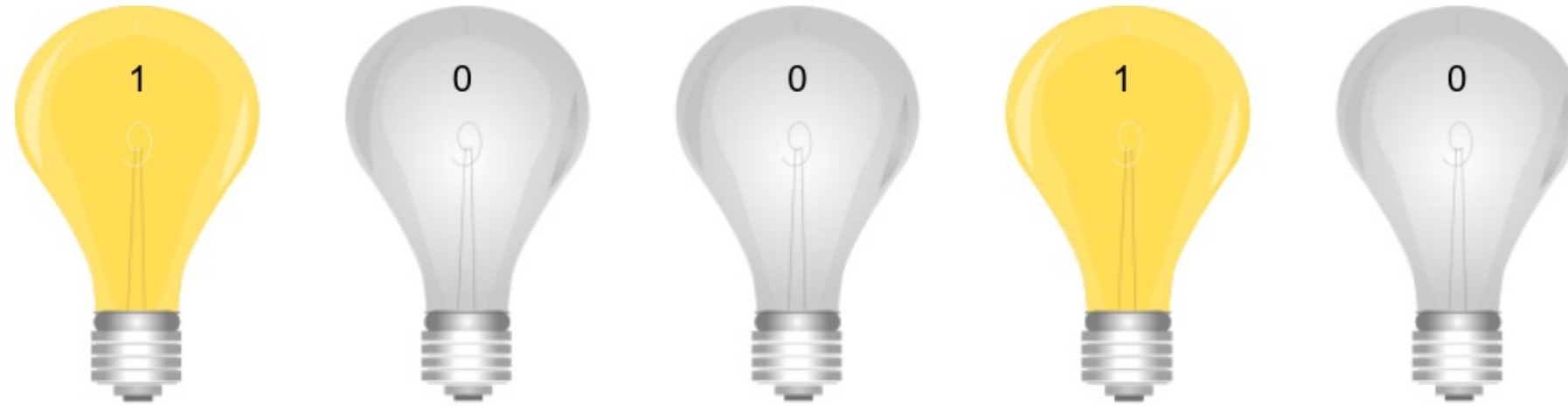
16

Representação de números: noção intuitiva do sistema binário



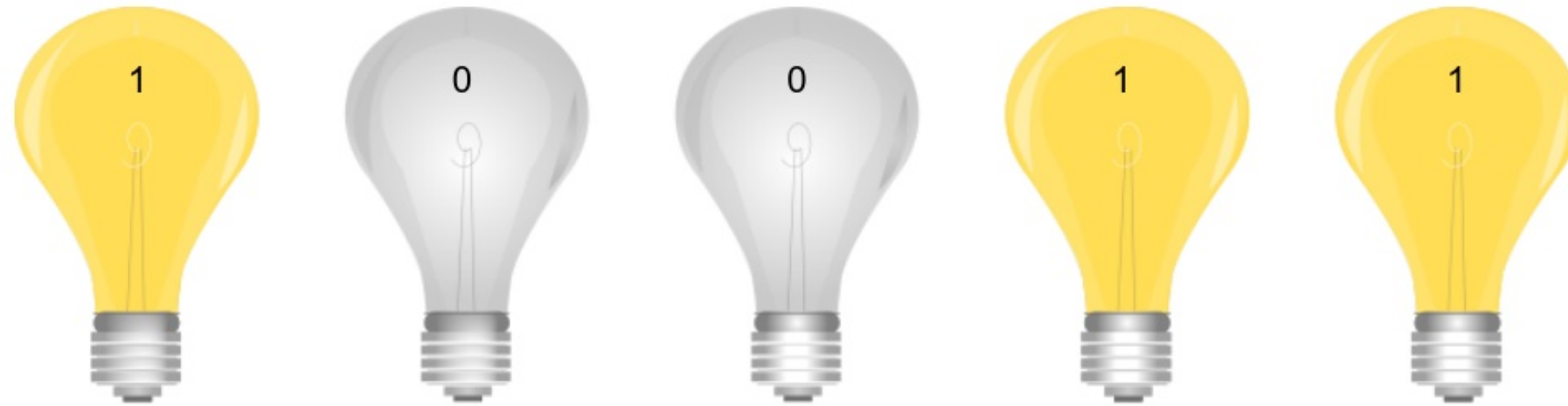
17

Representação de números: noção intuitiva do sistema binário



18

Representação de números: noção intuitiva do sistema binário



19

Representação de números: noção intuitiva do sistema binário



20

Representação de números: noção intuitiva do sistema binário



21

Representação de números: noção intuitiva do sistema binário



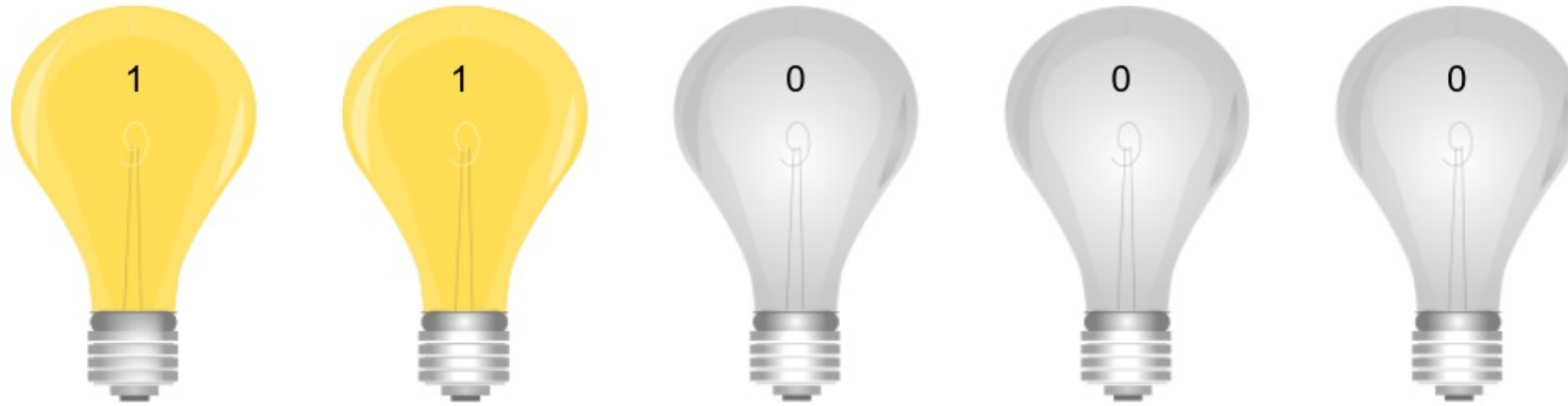
22

Representação de números: noção intuitiva do sistema binário



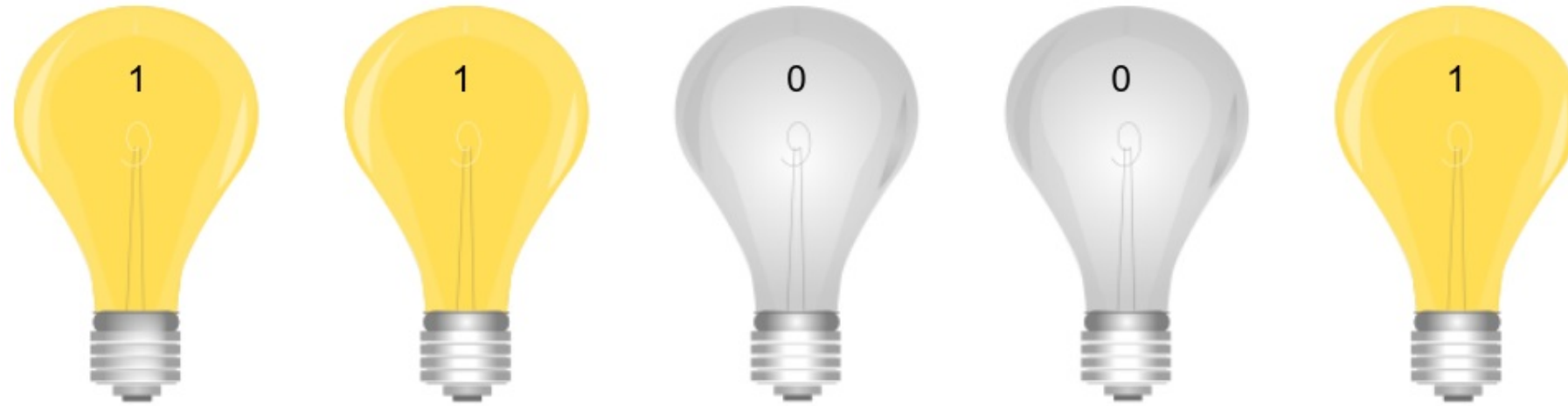
23

Representação de números: noção intuitiva do sistema binário



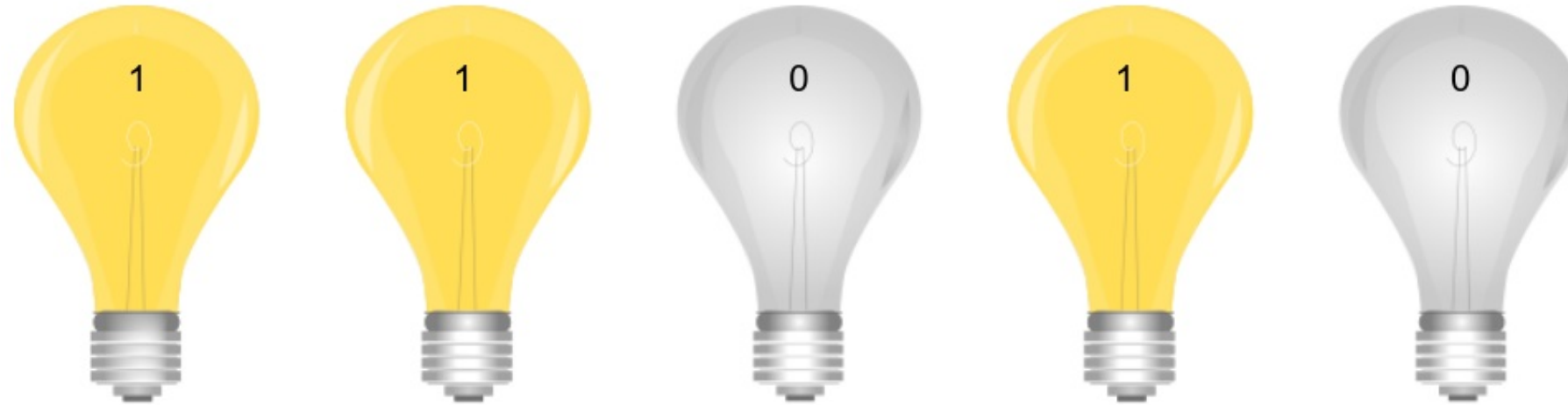
24

Representação de números: noção intuitiva do sistema binário



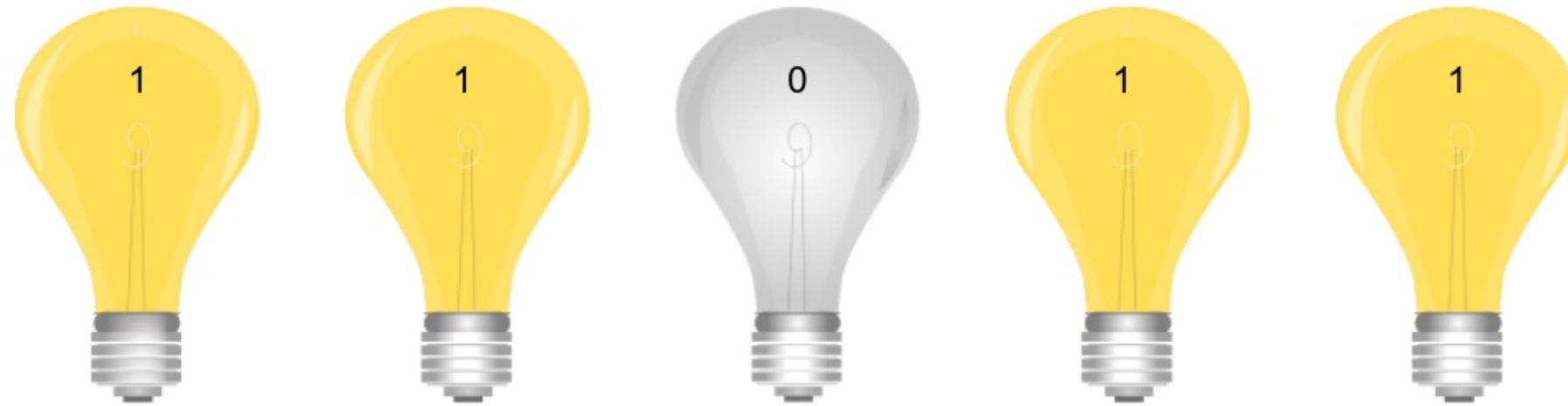
25

Representação de números: noção intuitiva do sistema binário



26

Representação de números: noção intuitiva do sistema binário



27

Representação de números: noção intuitiva do sistema binário



28

Representação de números: noção intuitiva do sistema binário



29

Representação de números: noção intuitiva do sistema binário



30

Representação de números: noção intuitiva do sistema binário



31

**Com 5 bits podemos representar 32 números, do 0 ao 31.
E se quiséssemos representar o número 32 ou o 33?**

Por que os computadores "falam" binário?

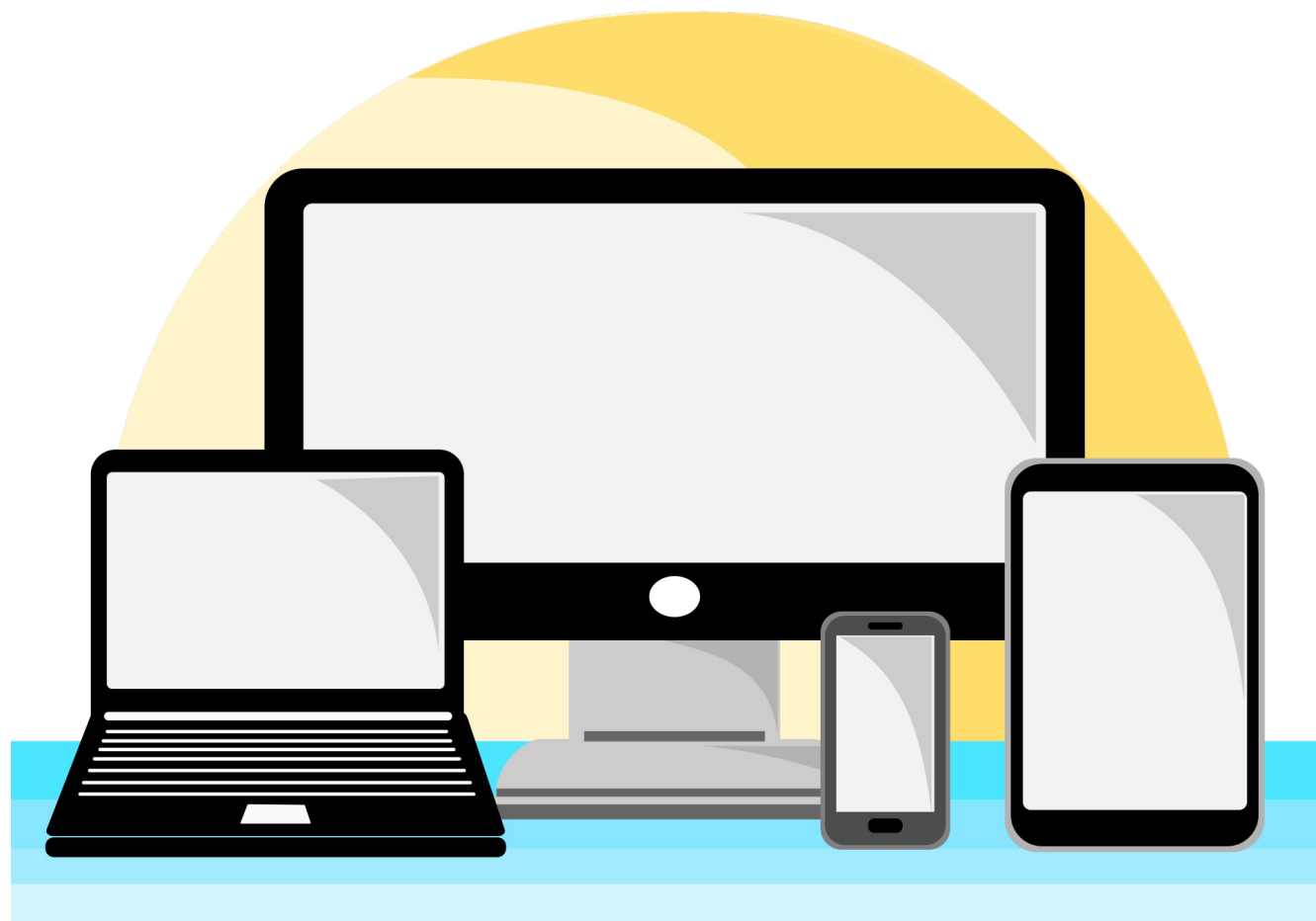


Imagem: JCamargo, no Pixabay (<https://pixabay.com/vectors/computers-cell-phone-notebook-2690565/>)

Imagem: OpenClipart-Vectors, no Pixabay
(<https://pixabay.com/vectors/button-on-off-power-engage-energy-161555/>)



sem energia
0

com energia
1

Por pura conveniência! É mais fácil representar apenas 2 algarismos (0 e 1) do que 10 algarismos!

Por que os computadores "falam" binário?

First Draft of a Report
on the EDVAC

by

John von Neumann

Contract No. W-670-ORD-4926

Between the

United States Army Ordnance Department

and the

University of Pennsylvania

"First Draft of a Report on the EDVAC"

30/06/1945

John von Neumann

Moore School of Electrical Engineering
University of Pennsylvania

June 30, 1945

5.0 Principles Governing the Arithmetical Operations

5.1 Let us now consider certain functions of the first specific part: the general arithmetical part CA.

The element in the sense of 4.3, the vacuum tube used as a current valve or gate, is an all-or-none device, or at least it approximates one: According to whether the grid bias is above or below cut-off, it will pass current or not. It is true that it needs definite potentials on all its electrodes in order to maintain either state, but there are combinations of vacuum tubes which have perfect equilibria: Several states in each of which the combination can exist indefinitely, without any outside support, while appropriate outside stimuli (electric pulses) will transfer it from one equilibrium into another. These are the so called trigger circuits, the basic one having two equilibria and containing two triodes or one pentode. The trigger circuits with more than two equilibria are disproportionately more involved.

Thus, whether the tubes are used as gates or as triggers, the all-or-none, two equilibrium arrangements are the simplest ones. Since these tube arrangements are to handle numbers by means of their digits, it is natural to use a system of arithmetic in which the digits are also two valued. This suggests the use of the binary system.

-14-

The analogs of human neurons, discussed in 4.2 - 4.3 are equally all-or-none elements. It will appear that they are quite useful for all preliminary, orienting considerations on vacuum tube systems (cf.). It is therefore satisfactory that here too, the natural arithmetical system to handle is the binary one.

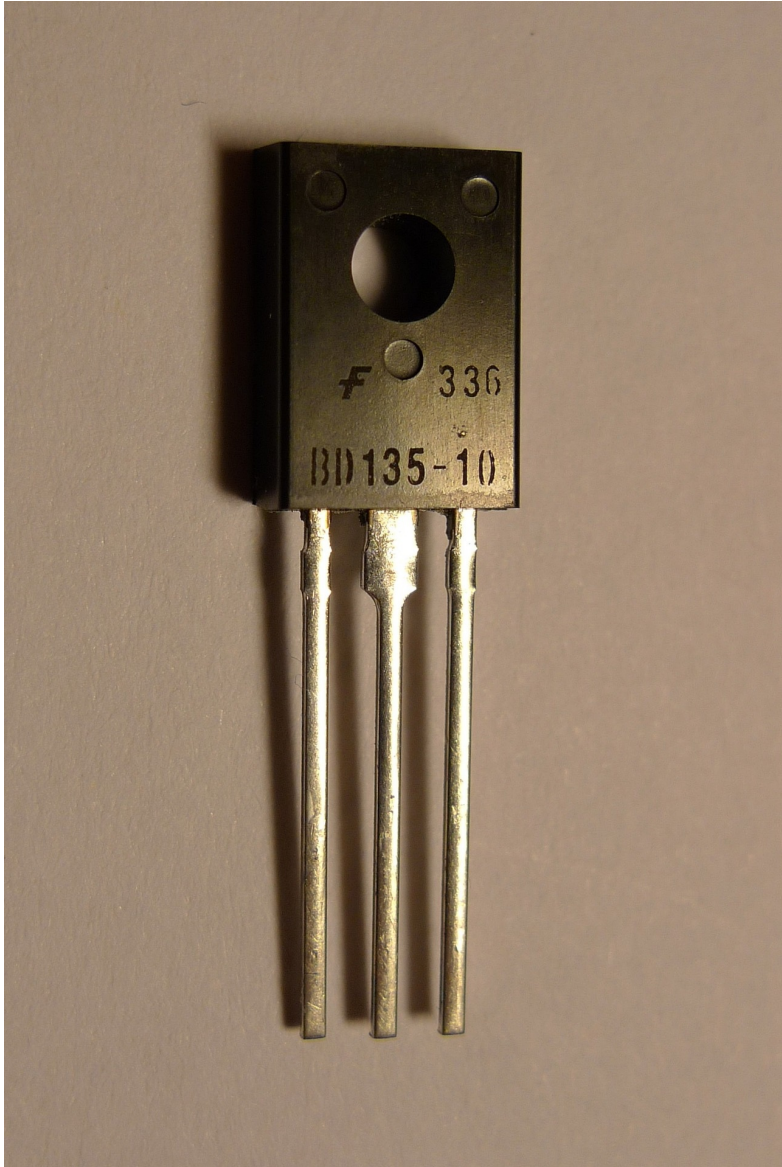
5.2 A consistent use of the binary system is also likely to simplify the operations of multiplication and division considerably. Specifically it does away with the decimal multiplication table, or with the alternative double procedure of building up the multiples by each multiplier or quotient digit by additions first, and then combining these (according to positional value) by a second sequence of additions or subtractions. In other words: Binary arithmetics has a simpler and more one-piece logical structure than any other, particularly than the decimal one.

Imagem: Smithsonian Libraries and Archives, no Internet Archives
(<https://archive.org/details/frstdraftofrepo00vonn/>)

"[...] O elemento no sentido de 4.3, o tubo de vácuo, usado como válvula ou portão de corrente, é um dispositivo tudo-ou-nada, ou pelo menos se aproxima de um: dependendo se a polarização do grid está acima ou abaixo de um nível de corte, passará corrente ou não. [...] Estes são os chamados circuitos gatilho, o mais básico tendo dois estados de equilíbrio [...]. Assim, quer os tubos sejam usados como portas ou como gatilhos, o tudo ou nada, o arranjo de dois equilíbrios é o mais simples. Como esses arranjos devem lidar com números por meio de seus dígitos, é natural usar um sistema de aritmética em que os dígitos também tenham dois valores. Isso sugere o uso do sistema binário. [...] O uso consistente do sistema binário também provavelmente simplificará as operações de multiplicação e divisão consideravelmente. Especificamente elimina a tabuada de multiplicação decimal [...] Em outras palavras: a aritmética binária tem uma estrutura lógica mais simples e completa do que qualquer outra, especialmente do que a decimal. [...]"



Quem "fabrica" os 0s e os 1s no computador?

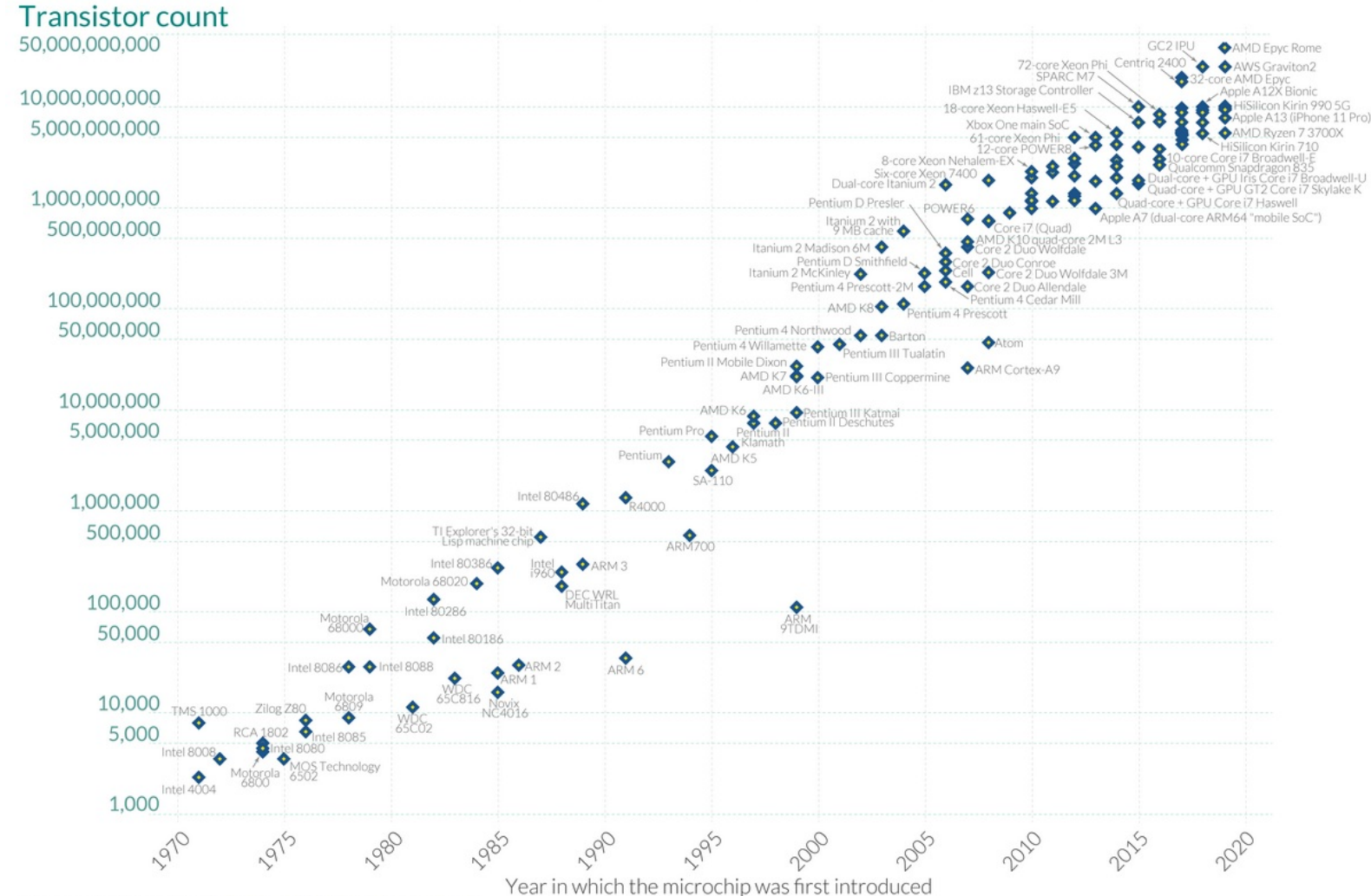


Transístor: responsável pelos 0s e 1s
0: quando interrompe a passagem de energia
1: quando permite a passagem de energia

Moore's Law: The number of transistors on microchips has doubled every two years



Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
 OurWorldinData.org – Research and data to make progress against the world's largest problems. Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

Imagem: WikimediaImages, no Pixabay
 (https://pixabay.com/photos/transistor-bd-135-electronic-903642/)

Imagem: Our World in Data (https://ourworldindata.org/moores-law)

Choque de realidade: os 0s e 1s NÃO EXISTEM! São uma ABSTRAÇÃO!



Imagem: OpenClipart-Vectors, no Pixabay
(<https://pixabay.com/vectors/button-on-off-power-engage-energy-161555/>)

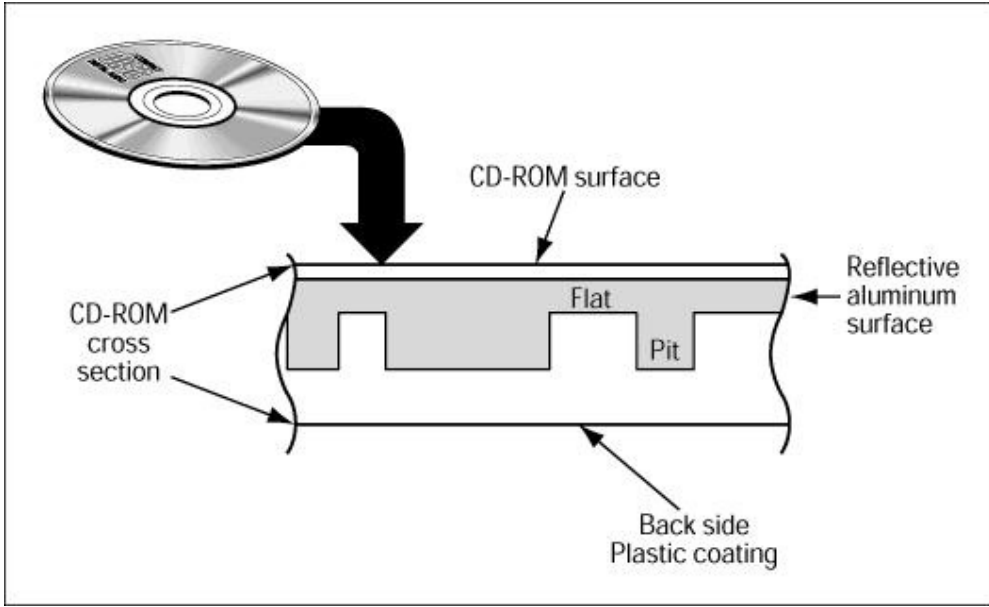


Imagem: Marko Medenica, Aaron Day e Natalia Burina, University of Washington
(<https://courses.cs.washington.edu/courses/cse370/01au/minirproject/370Leopards/cse370MiniResearch.htm>)

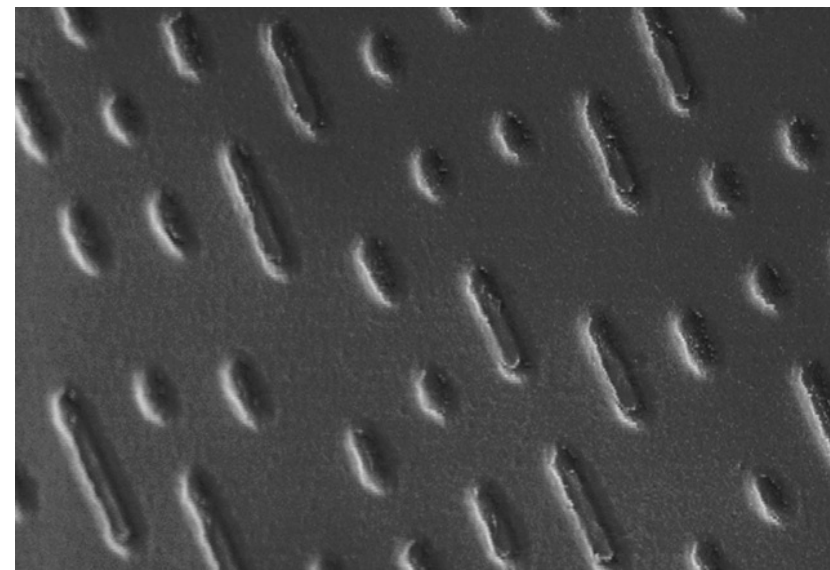


Imagem: Paul Murrel, Wirtschaftsuniversität Wien
(<https://statmath.wu.ac.at/courses/data-analysis/itdtHTML/node55.html>)

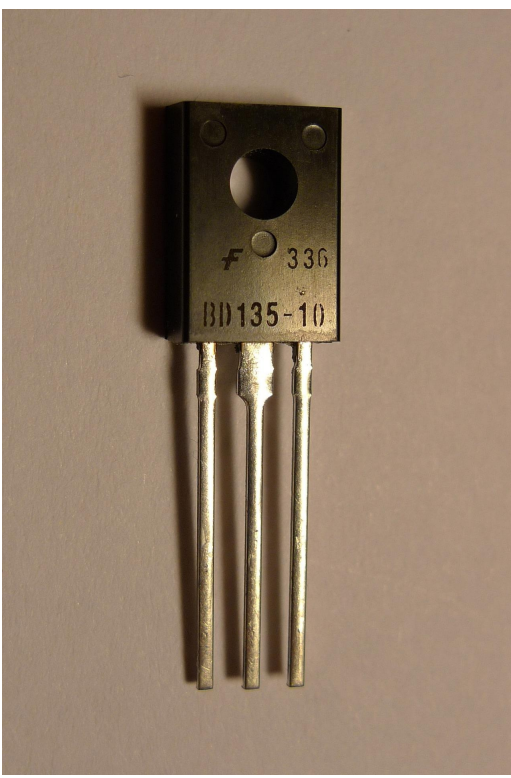


Imagem: WikimediaImages, no Pixabay
(<https://pixabay.com/photos/transistor-bd-135-electronic-903642/>)

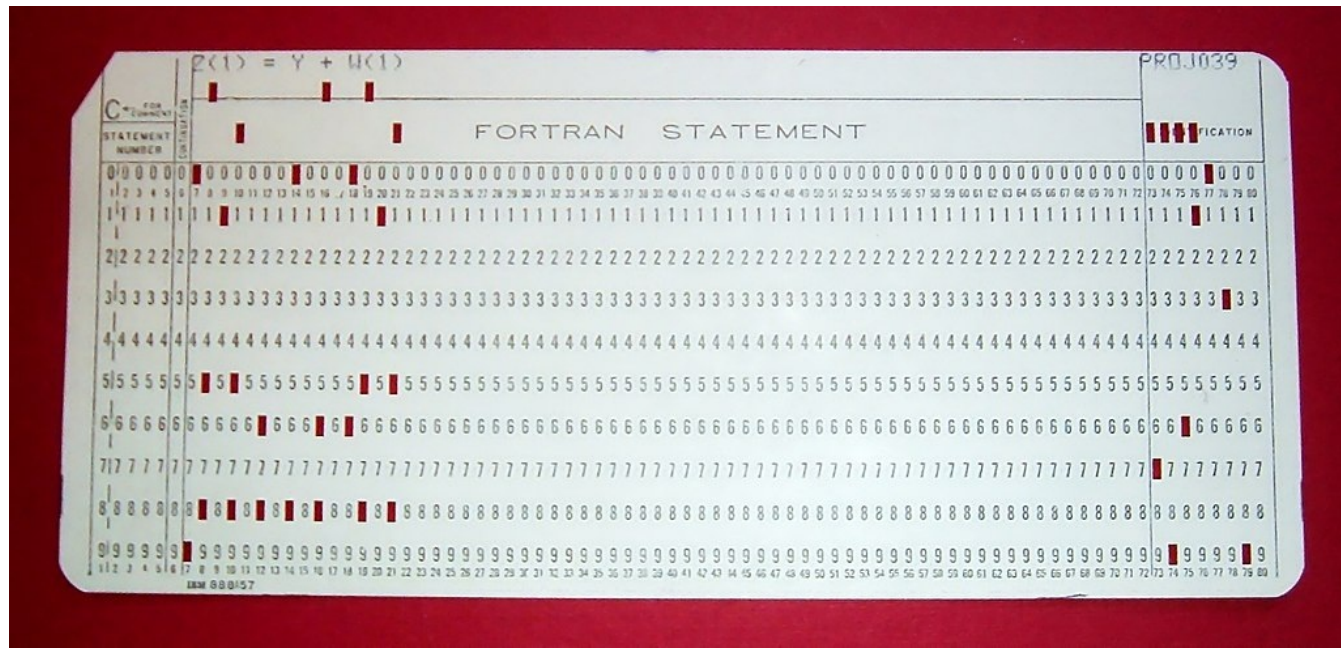


Imagem: Arnold Reinhold, na Wikimedia Commons
(<https://commons.wikimedia.org/wiki/File:FortranCardPROJ039.agr.jpg>)

Choque de realidade: os 0s e 1s NÃO EXISTEM! São uma ABSTRAÇÃO!

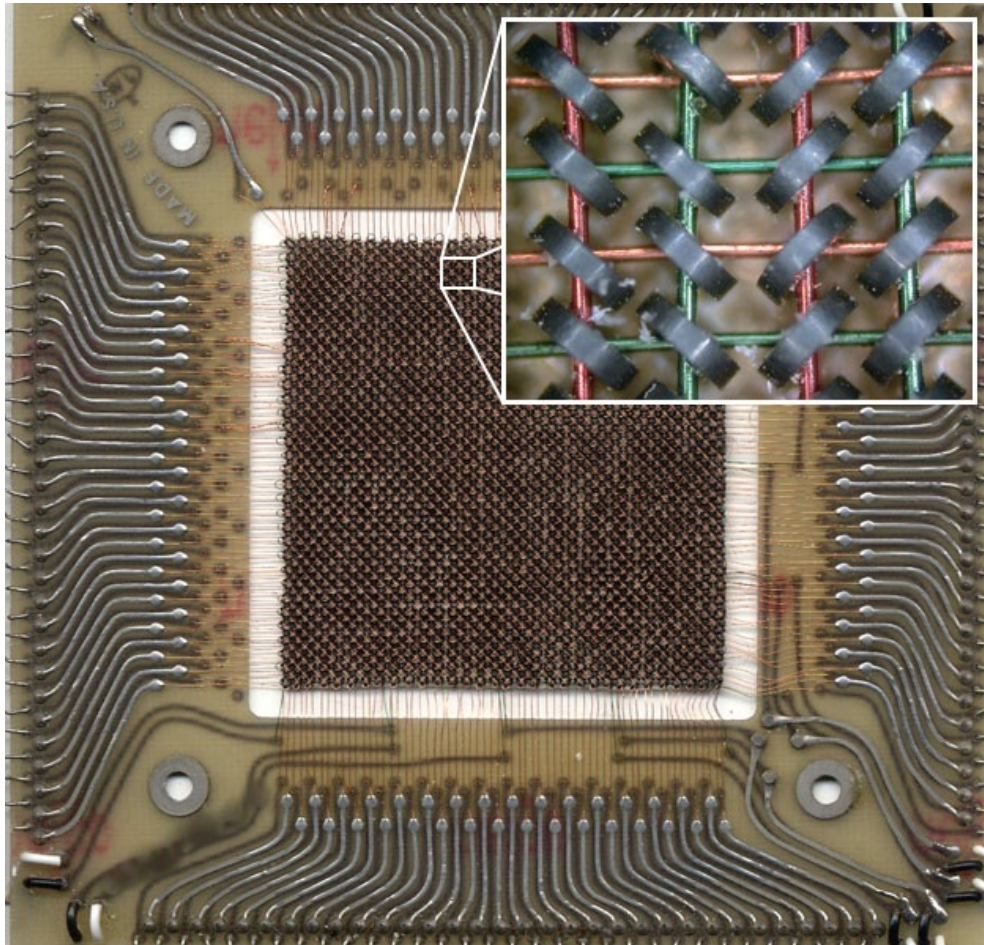


Imagem: Orion 8, na Wikipedia (https://en.wikipedia.org/wiki/File:Ferrite_core_memory.jpg)

**64 x 64 (4.096) bits de memória,
CDC 6600.**



Jitze Couperus, na Wikipedia (https://commons.wikimedia.org/wiki/File:CDC_6600.jc.jpg)

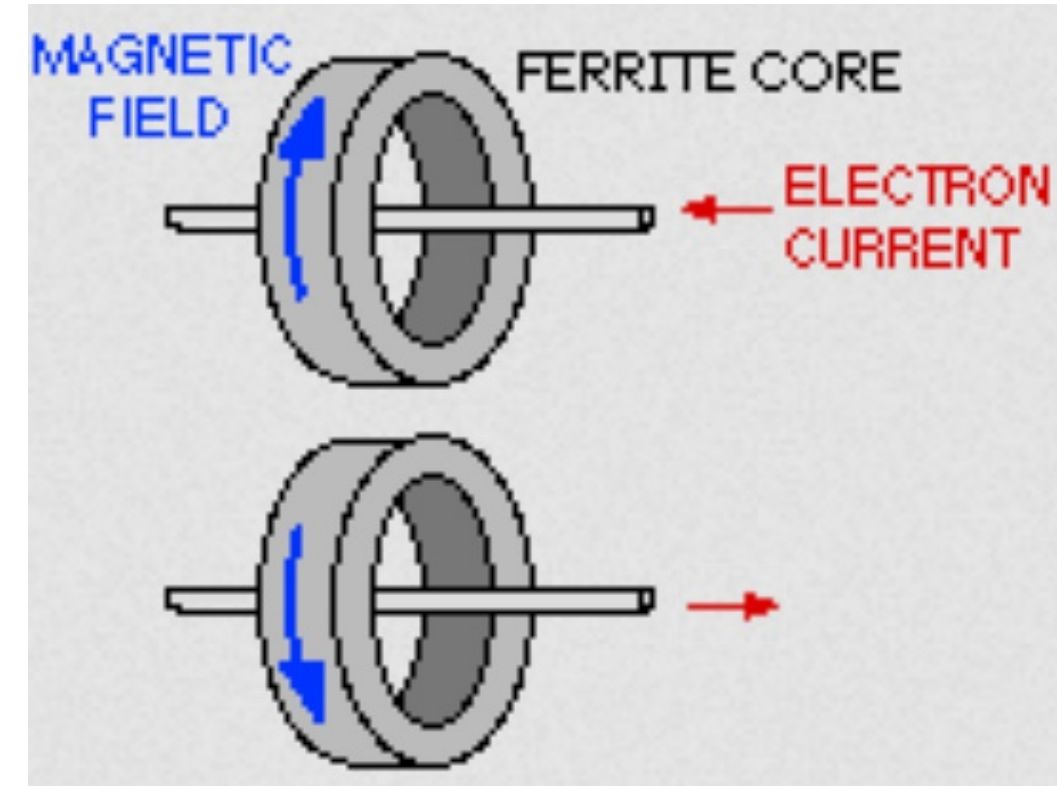
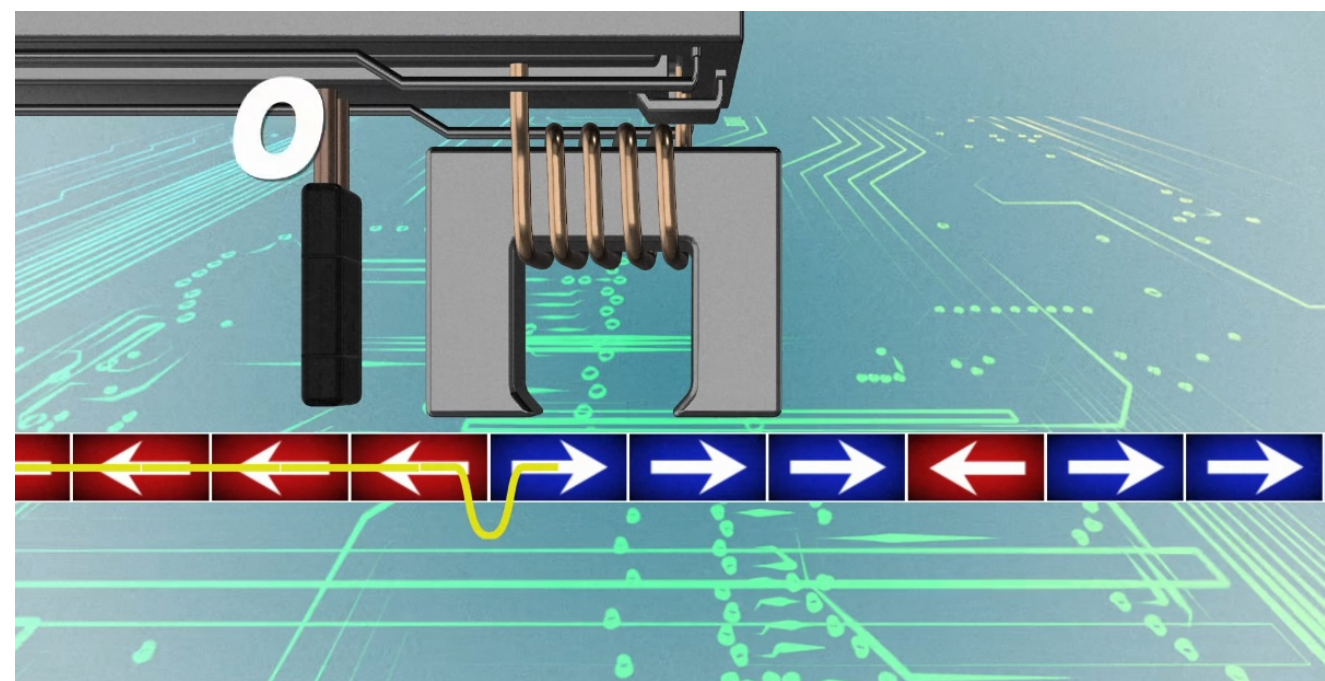
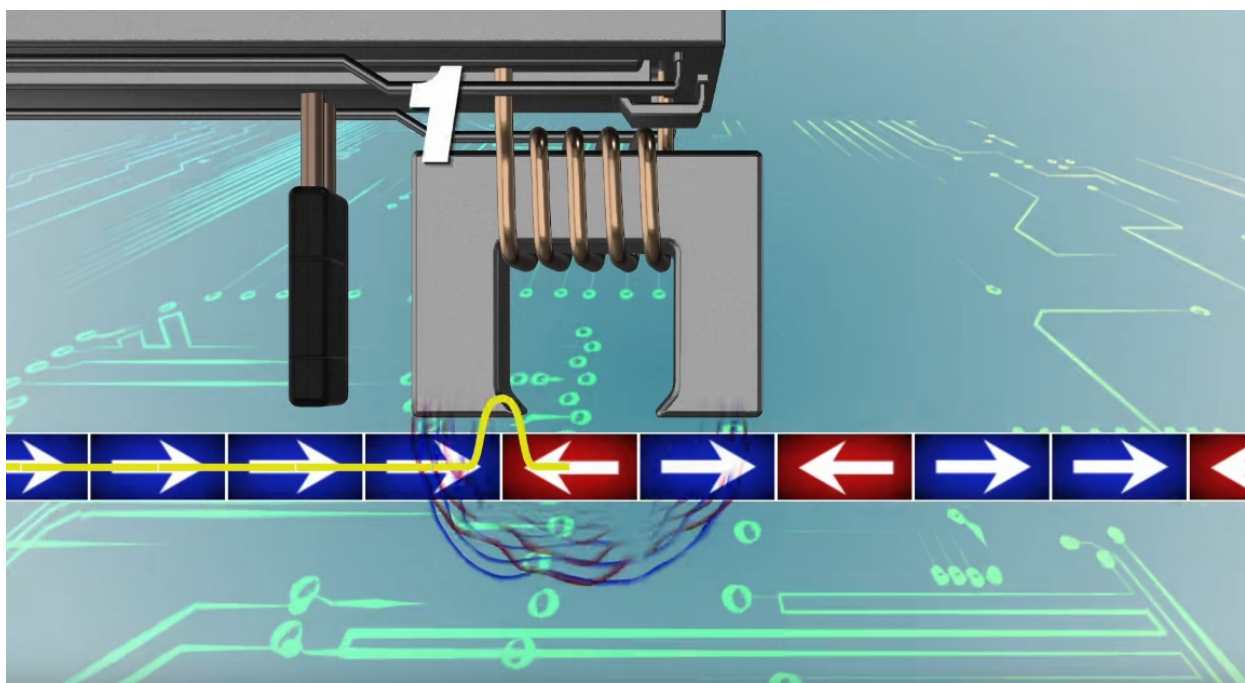
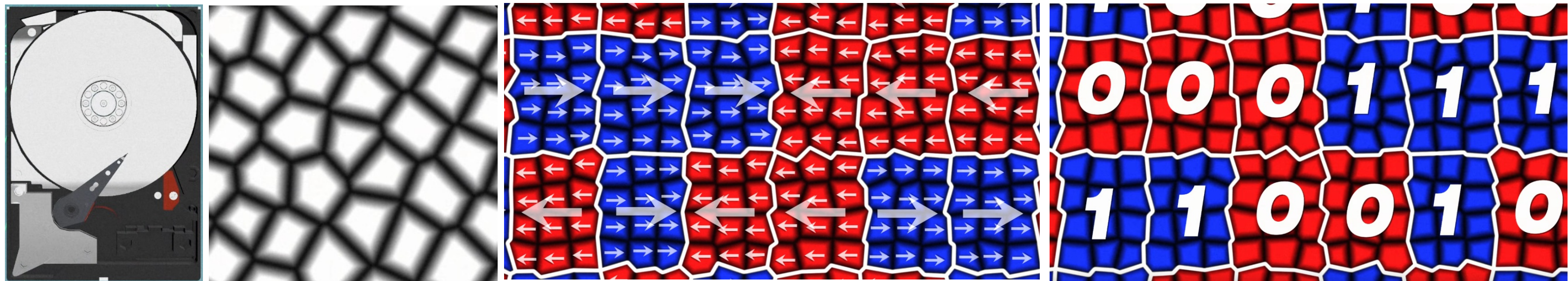


Imagem: adaptado de "Magnetic Core Memory Systems", por Brent Hilpert (<http://madrona.ca/e/coremem/index.html>)

Choque de realidade: os 0s e 1s NÃO EXISTEM! São uma ABSTRAÇÃO!



Imagens: prints criados a partir do vídeo "How do hard drives work?", de Kanawat Senanan, no YouTube (<https://www.youtube.com/watch?v=wteUW2sL7bc>)

Choque de realidade: os 0s e 1s NÃO EXISTEM! São uma ABSTRAÇÃO!

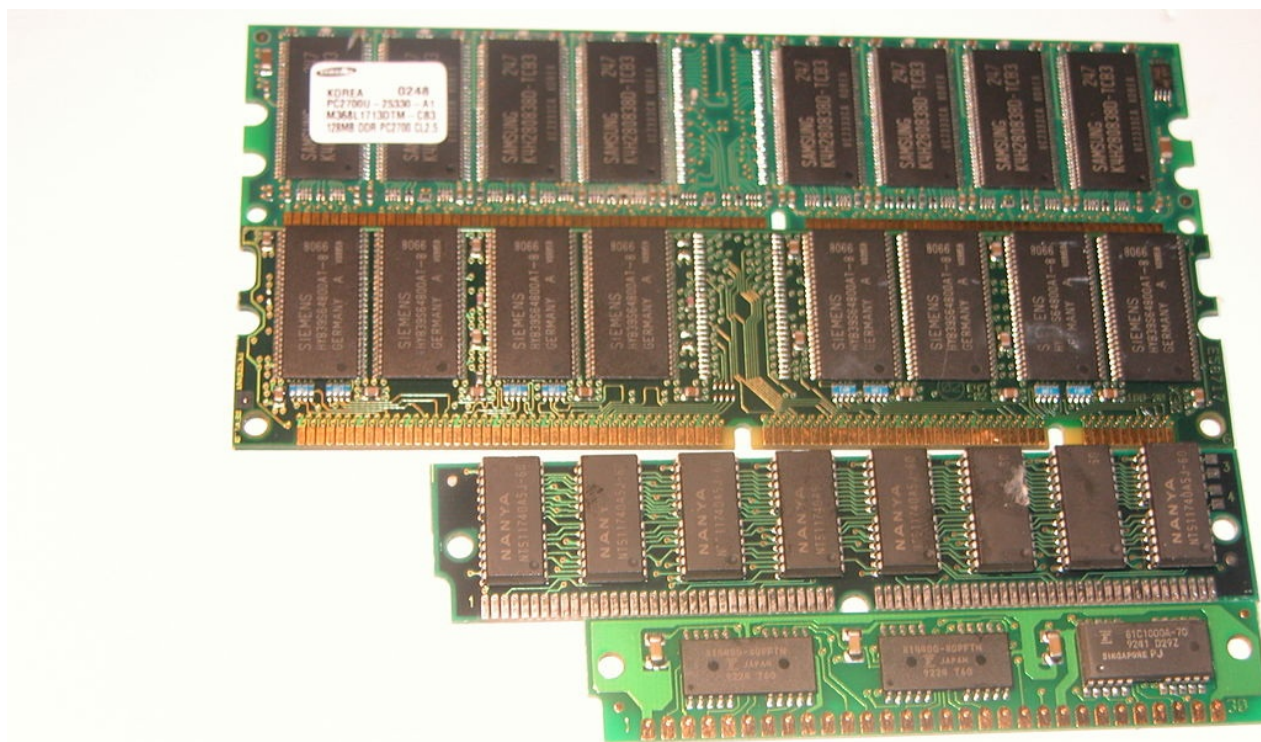


Imagem: Gyga, na Wikimedia Commons (<https://commons.wikimedia.org/wiki/File:Kinds-of-RAM.JPG>)

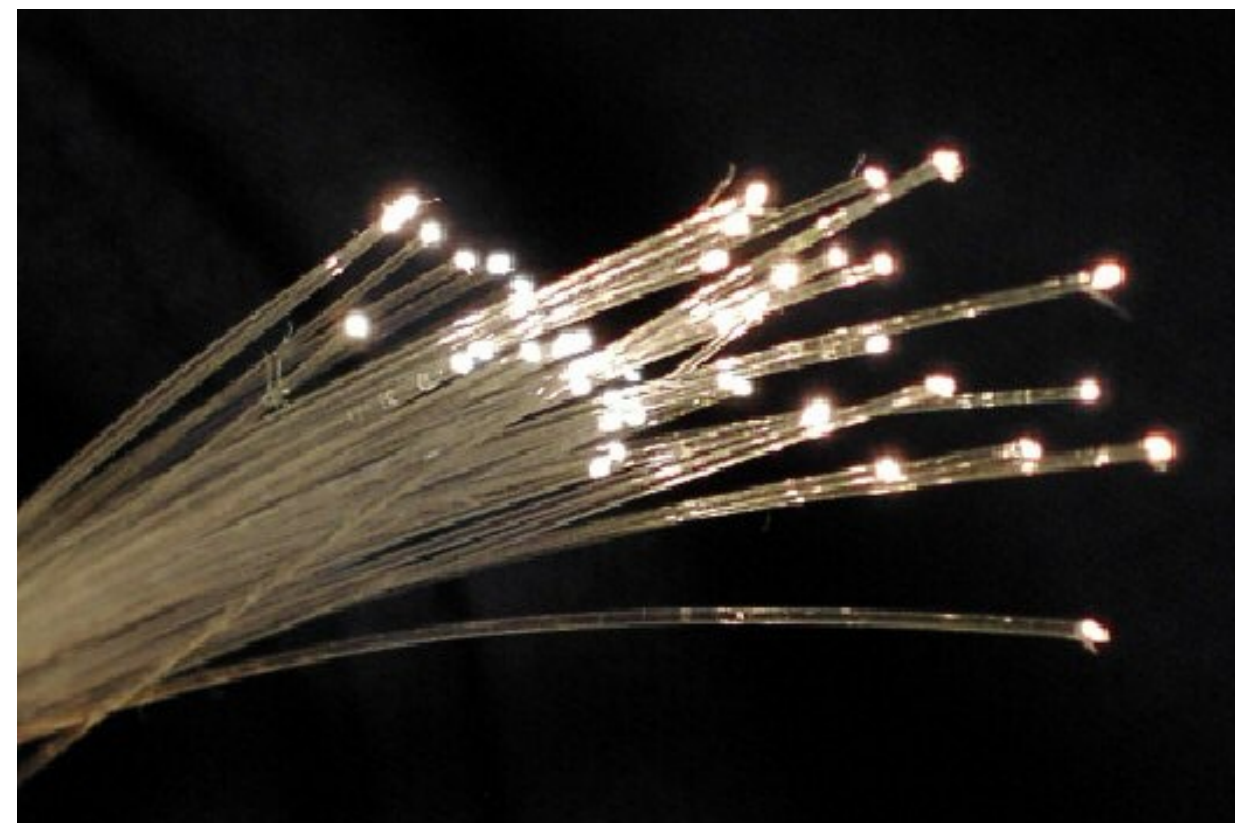


Imagem: BigRiz, na Wikimedia Commons (<https://commons.wikimedia.org/wiki/File:Fibreoptic.jpg>)

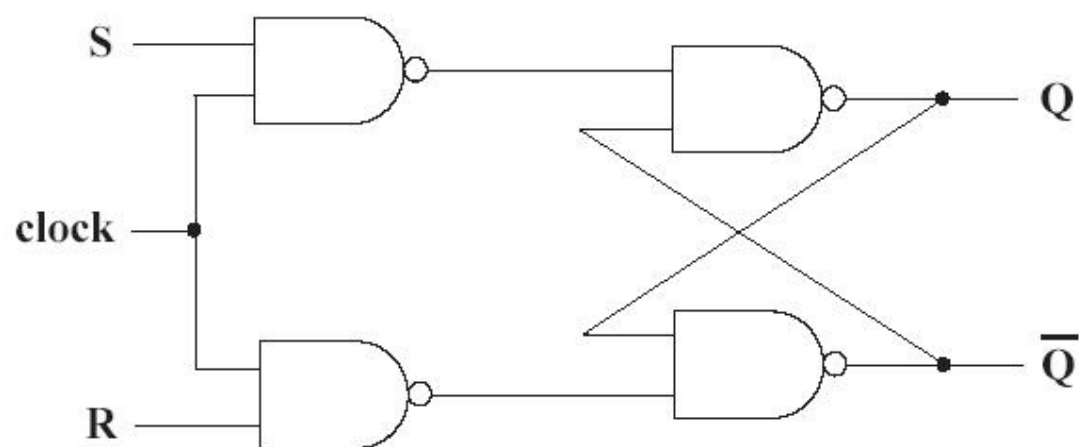


Imagem: Thyago Quintas, na Wikimedia Commons (https://commons.wikimedia.org/wiki/File:Flip-flop_SR.JPG)

Dispositivos **bi-estáveis** podem ser usados, desde que:

- a) os estados sejam separados;
- b) os estados podem ser lidos; e
- c) os estados podem ser alternados.

Até aqui: noção intuitiva

Encontrar uma representação adequada para a entrada e a saída de um problema é essencial:

- **Ao mudar a representação podemos fazer melhor**

Computadores falam binário:

- **Intuitivamente funciona como o hodômetro de um carro**
- **É a representação mais conveniente**
- **bit (binary digit): é cada algarismo 0 ou 1**

Os algarismos binários 0 e 1, no computador, são abstrações:

- **Eletricidade (transístores)**
- **Sulcos (CD-ROM)**
- **Direção do campo magnético (cores de memória, HD)**
- **Furos (cartão perfurado)**
- **Eletricidade (circuitos flip-flop)**
- **Luz (fibra óptica)**

Representação de números: formalização (a professora da 5ª série estava certa)



Imagem: 12019, no Pixabay (<https://pixabay.com/photos/school-classroom-boys-girls-79612/>)

Quando ela falava que você usaria a matéria no futuro, não estava brincando!

Para entender a representação de números no computador, você precisa relembrar:

- **Base de um sistema numérico** (quantidade de algarismos)
- **Valor posicional** de um algarismo
- **Decomposição** numérica

(sim, matéria da 5ª série)

Representação de números: sistema decimal

Base do sistema decimal: 10

Valor posicional de um algarismo: $n_i \times 10^i$

n = algarismo

i = posição (da direita para esquerda, inicia em 0)

n_i = algarismo na posição **i**

Decomposição numérica:

$$(n_i \times 10^i) + (n_{i-1} \times 10^{i-1}) + (n_{i-2} \times 10^{i-2}) + \dots + (n_0 \times 10^0)$$



Imagem: adaptado de atevern07, no Pixabay
(<https://pixabay.com/illustrations/balloon-foil-balloon-foil-number-7185735/>)

Representação de números: sistema decimal

$$(n_i \times 10^i) + (n_{i-1} \times 10^{i-1}) + (n_{i-2} \times 10^{i-2}) + \dots + (n_0 \times 10^0)$$

$$n_i \times 10^i$$

$$\begin{aligned} 364 &= (3 \times 10^2) + (6 \times 10^1) + (4 \times 10^0) \\ &= 300 + 60 + 4 \end{aligned}$$

Representação de números: sistema decimal

Quadro de Valor Posicional

Unidades de Trilhões			Unidades de Bilhões			Unidades de Milhões			Unidades de Milhar			Unidades Simples			← Classes
15 ^a	14 ^a	13 ^a	12 ^a	11 ^a	10 ^a	9 ^a	8 ^a	7 ^a	6 ^a	5 ^a	4 ^a	3 ^a	2 ^a	1 ^a	← Ordens
14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	← Posição
Centenas de trilhões	Dezenas de trilhões	Unidades de trilhões	Centenas de bilhões	Dezenas de bilhões	Unidades de bilhões	Centenas de milhões	Dezenas de milhões	Unidades de milhões	Centenas de milhar	Dezenas de milhar	Unidades de milhar	Centenas	Dezenas	Unidades	
								1	4	8	3	0	0	7	

$$(1 \times 10^6) + (4 \times 10^5) + (8 \times 10^4) + (3 \times 10^3) + (0 \times 10^2) + (0 \times 10^1) + (7 \times 10^0)$$

$$1.000.000 + 400.000 + 80.000 + 3.000 + 0 + 0 + 7$$

$$1.483.007$$

Representação de números: sistema binário

Base do sistema binário: 2

Valor posicional de um algarismo: $n_i \times 2^i$

n = algarismo

i = posição (da direita para esquerda, inicia em 0)

n_i = algarismo na posição **i**

Decomposição numérica:

$$(n_i \times 2^i) + (n_{i-1} \times 2^{i-1}) + (n_{i-2} \times 2^{i-2}) + \dots + (n_0 \times 2^0)$$



Imagem: adaptado de atevern07, no Pixabay
(<https://pixabay.com/illustrations/balloon-foil-balloon-foil-number-7185735/>)

Representação de números: sistema binário

$$(n_i \times 2^i) + (n_{i-1} \times 2^{i-1}) + (n_{i-2} \times 2^{i-2}) + \dots + (n_0 \times 2^0)$$

$$n_i \times 2^i$$

$$\begin{aligned} 10011 &= (1 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) \\ &= 16 + 0 + 0 + 2 + 1 \\ &= 19 \end{aligned}$$

Representação de números: sistema binário

Quadro de Valor Posicional

2								1								← Bytes
16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	← bits
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	← Posição
(32768) Trinta e dois mil setecentos e sessenta e oito																
(16384) Dezesesseis mil trezentos e oitenta e quatro																
(8192) Oito mil dento e noventa e dois																
(4096) Quatro mil e noventa e seis																
(2048) Dois mil e quarenta e oito																
(1024) Mil e vinte e quatro																
(512) Quinhentos e doze						1	0									
(256) Duzentos e cinquenta e seis						0										
(128) Cento e vinte e oito						0		0	0	0	1	0	1	0	1	
(64) Sessenta e quatro																
(32) Trinta e dois																
(16) Dezesesseis																
(8) Oito																
(4) Quatro																
(2) Dois																
(1) Um																

$$(1 \times 2^9) + (0 \times 2^8) + (0 \times 2^7) + (0 \times 2^6) + (0 \times 2^5) + (1 \times 2^4) + (0 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$$

$$512 + 0 + 0 + 0 + 0 + 16 + 0 + 4 + 0 + 1$$

$$533$$

Representação de números: sistema octal

Base do sistema octal: 8

Valor posicional de um algarismo: $n_i \times 8^i$

n = algarismo

i = posição (da direita para esquerda, inicia em 0)

n_i = algarismo na posição i

Decomposição numérica:

$$(n_i \times 8^i) + (n_{i-1} \times 8^{i-1}) + (n_{i-2} \times 8^{i-2}) + \dots + (n_0 \times 8^0)$$



Imagem: adaptado de atevern07, no Pixabay
(<https://pixabay.com/illustrations/balloon-foil-balloon-foil-number-7185735/>)

Representação de números: sistema octal

$$(n_i \times 8^i) + (n_{i-1} \times 8^{i-1}) + (n_{i-2} \times 8^{i-2}) + \dots + (n_0 \times 8^0)$$

$$n_i \times 8^i$$

$$\begin{aligned} 17034 &= (1 \times 8^4) + (7 \times 8^3) + (0 \times 8^2) + (3 \times 8^1) + (4 \times 8^0) \\ &= 4096 + 3584 + 0 + 24 + 4 \\ &= 7708 \end{aligned}$$

Representação de números: sistema octal

Quadro de Valor Posicional

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	← Posição
(35.184.372.088.832) Trinta e cinco trilhões cento ...	(4.398.046.511.104) Quatro trilhões trezentos e ...	(549.755.813.888) Quinhentos e quarenta e nove ...	(68.719.476.736) Sessenta e oito bilhões setecentos ...	(8.589.934.592) Oito bilhões quinhentos e oitenta...	(1.073.741.824) Um bilhão setenta e três milhões...	(134.217.728) Cento e trinta e quatro milhões ...	(16.777.216) Dezesesseis milhões setecentos e setenta...	(2.097.152) Dois milhões noventa e sete mil cento ...	(262.144) Duzentos e sessenta e dois mil cento e ...	(32.768) Trinta e dois mil setecentos e sessenta e ...	(4.096) Quatro mil e noventa e seis	(512) Quinhentos e doze	(64) Sessenta e quatro	(8) Oito	(1) Um	
							5	7	0	0	3	4	1	5	3	

$$(5 \times 8^8) + (7 \times 8^7) + (0 \times 8^6) + (0 \times 8^5) + (3 \times 8^4) + (4 \times 8^3) + (1 \times 8^2) + (5 \times 8^1) + (3 \times 8^0)$$

$$86.889.080 + 14.680.064 + 0 + 0 + 12.288 + 2.048 + 64 + 40 + 3$$

$$98.580.587$$

Representação de números: sistema hexadecimal

Base do sistema hexadecimal: 16

Valor posicional de um algarismo: $n_i \times 16^i$

n = algarismo

i = posição (da direita para esquerda, inicia em 0)

n_i = algarismo na posição **i**

Decomposição numérica:

$$(n_i \times 16^i) + (n_{i-1} \times 16^{i-1}) + (n_{i-2} \times 16^{i-2}) + \dots + (n_0 \times 16^0)$$

Hexadecimal	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

Representação de números: sistema hexadecimal

$$(n_i \times 16^i) + (n_{i-1} \times 16^{i-1}) + (n_{i-2} \times 16^{i-2}) + \dots + (n_0 \times 16^0)$$

$$n_i \times 16^i$$

$$\begin{aligned} \text{F90B4} &= (15 \times 16^4) + (9 \times 16^3) + (0 \times 16^2) + (11 \times 16^1) + (4 \times 16^0) \\ &= 983040 + 36864 + 0 + 176 + 4 \\ &= 1020084 \end{aligned}$$

Hexadecimal	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

Representação de números: sistema hexadecimal

Quadro de Valor Posicional

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	← Posição
(1.152.921.504.606.850.000) Um quintilhão, cento e...	(72.057.594.037.927.900) Setenta e dois quatrilhões...	(4.503.599.627.370.496) Quatro quatrilhões quinhentos...	(281.474.976.710.656) Duzentos e oitenta e um trilhões...	(17.592.186.044.416) Dezesete trilhões quinhentos ...	(1.099.511.627.776) Um trilhão noventa e nove bilhões...	(68.719.476.736) Sessenta e oito bilhões setecentos ...	(4.294.967.296) Quatro bilhões duzentos e noventa...	(268.435.456) Duzentos e sessenta e oito milhões ...	(16.777.216) Dezesesseis milhões setecentos e setenta ...	(1.048.576) Um milhão quarenta e oito mil quinhentos...	(65.536) Sessenta e cinco mil quinhentos e trinta ...	(4.096) Quatro mil e noventa e seis	(256) Duzentos e cinquenta e seis	(16) Dezesesseis	(1) Um	
							1	B	0	0	0	0	A	9	F	

$$(1 \times 16^8) + (11 \times 16^7) + (0 \times 16^6) + (0 \times 16^5) + (0 \times 16^4) + (0 \times 16^3) + (10 \times 16^2) + (9 \times 16^1) + (14 \times 16^0)$$

$$4.294.967.296 + 2.952.790.016 + 0 + 0 + 0 + 0 + 2.560 + 144 + 14$$

$$7.247.760.030$$

Hexadecimal	Decimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

Representação de números: generalização

Base do sistema: b

Valor posicional de um algarismo: $n_i \times b^i$

n = algarismo

i = posição (da direita para esquerda, inicia em 0)

n_i = algarismo na posição i

Decomposição numérica:

$$(n_i \times b^i) + (n_{i-1} \times b^{i-1}) + (n_{i-2} \times b^{i-2}) + \dots + (n_0 \times b^0)$$

Representação de números: unidades de medida decimal

Tabela 1: Grandezas de base 10 maiores do que a unidade

Fator	Prefixo	Símbolo	Número	Nome
10^{32}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de nonilhão
10^{31}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de nonilhão
10^{30}	quetta	Q	1 000 000 000 000 000 000 000 000 000 000	Nonilhão
10^{29}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de octilhão
10^{28}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de octilhão
10^{27}	ronna	R	1 000 000 000 000 000 000 000 000 000 000	Octilhão
10^{26}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de septilhão
10^{25}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de septilhão
10^{24}	yotta	Y	1 000 000 000 000 000 000 000 000 000 000	Septilhão
10^{23}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de sextilhão
10^{22}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de sextilhão
10^{21}	zetta	Z	1 000 000 000 000 000 000 000 000 000 000	Sextilhão
10^{20}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de quintilhão
10^{19}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de quintilhão
10^{18}	exa	E	1 000 000 000 000 000 000 000 000 000 000	Quintilhão
10^{17}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de quatrilhão
10^{16}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de quatrilhão
10^{15}	peta	P	1 000 000 000 000 000 000 000 000 000 000	Quatrilhão
10^{14}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de trilhão
10^{13}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de trilhão
10^{12}	tera	T	1 000 000 000 000 000 000 000 000 000 000	Trilhão
10^{11}	-	-	100 000 000 000 000 000 000 000 000 000	Centena de bilhão
10^{10}	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de bilhão
10^9	giga	G	1 000 000 000 000 000 000 000 000 000 000	Bilhão
10^8	-	-	100 000 000 000 000 000 000 000 000 000	Centena de milhão
10^7	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de milhão
10^6	mega	M	1 000 000 000 000 000 000 000 000 000 000	Milhão
10^5	-	-	100 000 000 000 000 000 000 000 000 000	Centena de milhar
10^4	-	-	10 000 000 000 000 000 000 000 000 000 000	Dezena de milhar
10^3	kilo	k	1 000 000 000 000 000 000 000 000 000 000	Milhar
10^2	hecto	h	100 000 000 000 000 000 000 000 000 000	Centena
10^1	deca	da	10 000 000 000 000 000 000 000 000 000 000	Dezena
10^0	-	-	1 000 000 000 000 000 000 000 000 000 000	Unidade

Tabela 2: Grandezas de base 10 menores do que a unidade

Fator	Prefixo	Símbolo	Número	Nome
10^0	-	-	1	Unidade
10^{-1}	deci	d	0,1	Décimo
10^{-2}	centi	c	0,01	Centésimo
10^{-3}	mili	m	0,001	Milésimo
10^{-4}	-	-	0,0001	Décimo de milésimo
10^{-5}	-	-	0,00001	Centésimo de milésimo
10^{-6}	micro	μ	0,000001	Milionésimo
10^{-7}	-	-	0,0000001	Décimo de milionésimo
10^{-8}	-	-	0,00000001	Centésimo de milionésimo
10^{-9}	nano	n	0,000000001	Bilionésimo
10^{-10}	-	-	0,0000000001	Décimo de bilionésimo
10^{-11}	-	-	0,00000000001	Centésimo de bilionésimo
10^{-12}	pico	p	0,000000000001	Trilionésimo
10^{-13}	-	-	0,0000000000001	Décimo de trilionésimo
10^{-14}	-	-	0,00000000000001	Centésimo de trilionésimo
10^{-15}	femto	f	0,000000000000001	Quatrilionésimo
10^{-16}	-	-	0,0000000000000001	Décimo de quatrilionésimo
10^{-17}	-	-	0,00000000000000001	Centésimo de quatrilionésimo
10^{-18}	atto	a	0,000000000000000001	Quintilionésimo
10^{-19}	-	-	0,0000000000000000001	Décimo de quintilionésimo
10^{-20}	-	-	0,00000000000000000001	Centésimo de quintilionésimo
10^{-21}	zepto	z	0,000000000000000000001	Sextilionésimo
10^{-22}	-	-	0,0000000000000000000001	Décimo de sextilionésimo
10^{-23}	-	-	0,00000000000000000000001	Centésimo de sextilionésimo
10^{-24}	yocto	y	0,000000000000000000000001	Septilionésimo
10^{-25}	-	-	0,0000000000000000000000001	Décimo de septilionésimo
10^{-26}	-	-	0,00000000000000000000000001	Centésimo de septilionésimo
10^{-27}	ronto	r	0,000000000000000000000000001	Octilionésimo
10^{-28}	-	-	0,0000000000000000000000000001	Décimo de octilionésimo
10^{-29}	-	-	0,00000000000000000000000000001	Centésimo de octilionésimo
10^{-30}	quecto	q	0,000000000000000000000000000001	Nonilionésimo
10^{-31}	-	-	0,0000000000000000000000000000001	Décimo de nonilionésimo
10^{-32}	-	-	0,00000000000000000000000000000001	Centésimo de nonilionésimo

Representação de números: unidades de medida decimal

Tabela 3: Resumo de ordens de grandeza padronizadas no SI

Fator	Prefixo	Símbolo	Nome
10^{30}	quetta	Q	Nonilhão
10^{27}	ronna	R	Octilhão
10^{24}	yotta	Y	Septilhão
10^{21}	zetta	Z	Sextilhão
10^{18}	exa	E	Quintilhão
10^{15}	peta	P	Quatrilhão
10^{12}	tera	T	Trilhão
10^9	giga	G	Bilhão
10^6	mega	M	Milhão
10^3	kilo	k	Milhar
10^2	hecto	h	Centena
10^1	deca	da	Dezena
10^0	-	-	Unidade
10^{-1}	deci	d	Décimo
10^{-2}	centi	c	Centésimo
10^{-3}	mili	m	Milésimo
10^{-6}	micro	μ	Milionésimo
10^{-9}	nano	n	Bilionésimo
10^{-12}	pico	p	Trilionésimo
10^{-15}	femto	f	Quatrilionésimo
10^{-18}	atto	a	Quintilionésimo
10^{-21}	zepto	z	Sextilionésimo
10^{-24}	yocto	y	Septilionésimo
10^{-27}	ronto	r	Octilionésimo
10^{-30}	quecto	q	Nonilionésimo

Representação de números: unidades de medida binária

bit	b	Um algarismo binário (0 ou 1)
nibble		Conjunto de 4 bits
Byte	B	Conjunto de 8 bits

Unidade fundamental de medida nos computadores.

Armazena, em geral, 1 caractere.

Tabela 4: Ordens de grandeza para potências de base 2

Fator	Prefixo	Símbolo
2^0	-	-
2^{10}	kibi	Ki
2^{20}	mebi	Mi
2^{30}	gibi	Gi
2^{40}	tebi	Ti
2^{50}	pebi	Pi
2^{60}	exbi	Ei
2^{70}	zebi	Zi
2^{80}	yobi	Yi
2^{90}	robi	Ri
2^{100}	quebi	Qi

Na Bíblia existem 3.566.480 letras (fonte: biblia.com.br). Para armazenar a Bíblia eu precisaria de quanto de espaço de armazenamento? Aprox. 3,6 MiB.

Um pendrive de 16 GiB armazenaria quantas Bíblias? Aprox. 4.550 Bíblias.

Representação de números: unidades decimais x unidades binárias

Comparação entre unidades decimais e binárias

Decimal			
	Fator	Prefixo (símbolo)	Valor
Unidade	10^0		1
Milhar	10^3	kilo (k)	1.000
Milhão	10^6	mega (M)	1.000.000
Bilhão	10^9	giga (G)	1.000.000.000
Trilhão	10^{12}	tera (T)	1.000.000.000.000
Quatrilhão	10^{15}	peta (P)	1.000.000.000.000.000
Quintilhão	10^{18}	exa (E)	1.000.000.000.000.000.000
Sextilhão	10^{21}	zetta (Z)	1.000.000.000.000.000.000.000
Septilhão	10^{24}	yotta (Y)	1.000.000.000.000.000.000.000.000
Octilhão	10^{27}	ronna (R)	1.000.000.000.000.000.000.000.000.000
Nonilhão	10^{30}	quetta (Q)	1.000.000.000.000.000.000.000.000.000.000

Binário			
	Fator	Prefixo (símbolo)	Valor
Unidade	2^0		1
Milhar	2^{10}	kibi (Ki)	1.024
Milhão	2^{20}	mebi (Mi)	1.048.576
Bilhão	2^{30}	gibi (Gi)	1.073.741.824
Trilhão	2^{40}	tebi (Ti)	1.099.511.627.776
Quatrilhão	2^{50}	pebi (Pi)	1.125.899.906.842.624
Quintilhão	2^{60}	exbi (Ei)	1.152.921.504.606.846.976
Sextilhão	2^{70}	zebi (Zi)	1.180.591.620.717.411.303.424
Septilhão	2^{80}	yobi (Yi)	1.208.925.819.614.629.174.706.176
Octilhão	2^{90}	robi (Ri)	1.237.940.039.285.380.274.899.124.224
Nonilhão	2^{100}	quebi (Qi)	1.267.650.600.228.229.401.496.703.205.376

Transformações, números negativos e frações: no anexo (importante!)

Método da divisão:

Decimal para Binário

Decimal para Octal

Decimal para Hexadecimal

Método indireto via binário:

Octal para Hexadecimal

Hexadecimal para Octal

Decomposição numérica:

Binário para Decimal

Octal para Decimal

Hexadecimal para Decimal

Números negativos

Agrupamento:

Binário para Octal

Binário para Hexadecimal

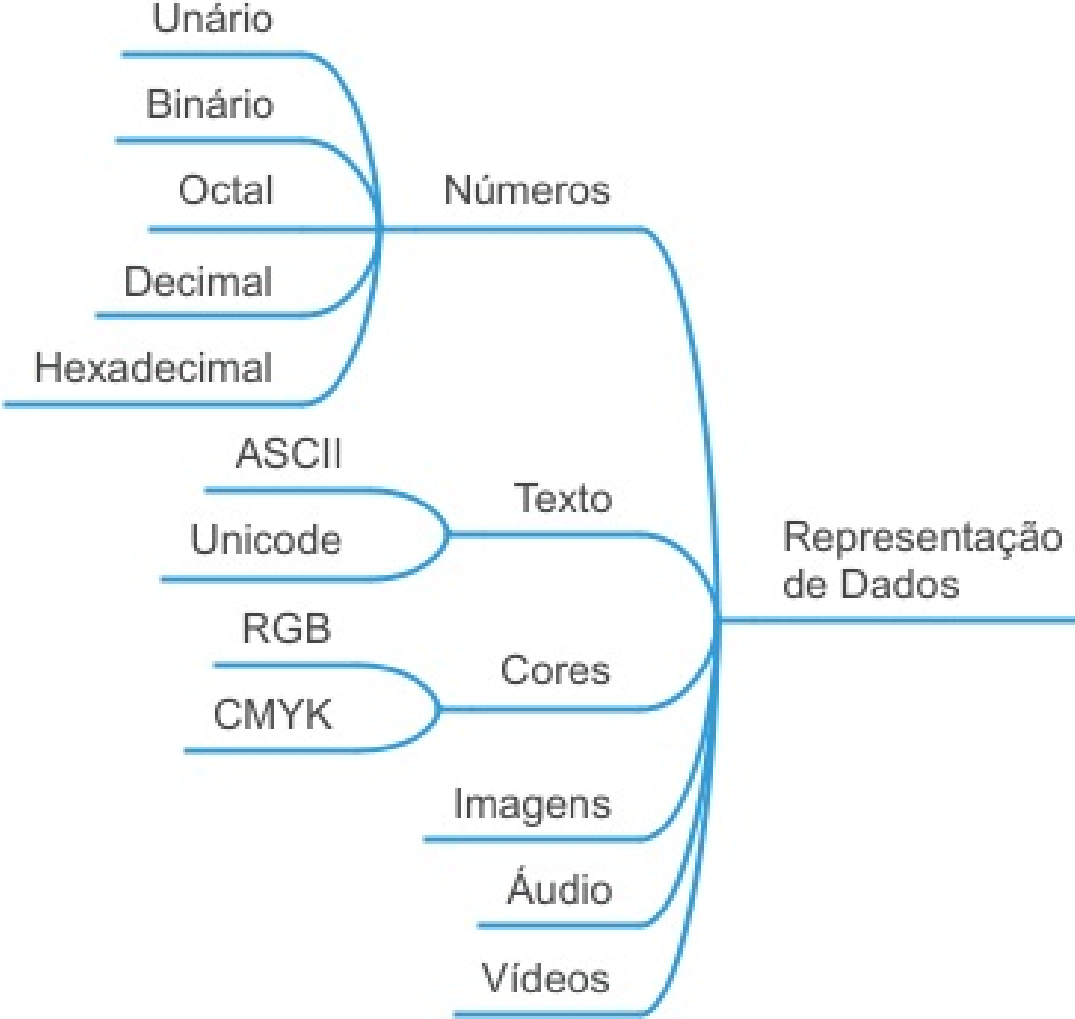
Números fracionários

Desagrupamento:

Octal para Binário

Hexadecimal para Binário

Representação de dados: texto



Encoding

ASCII

Sistemas de escrita e scripts

Unicode

- Code points
- Emojis

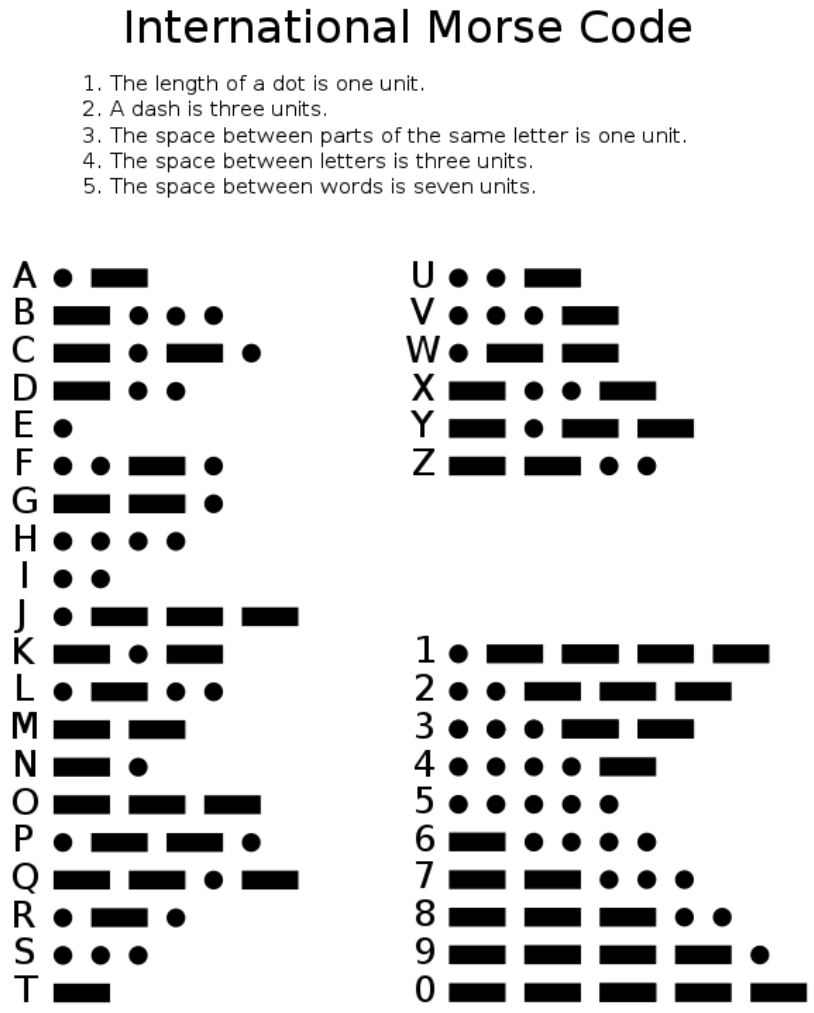
Como representar letras se tudo o que o computador entende são 0s e 1s?

Oi!

Com encodings!

Encoding: processo de mapear os caracteres em números binários, permitindo que eles sejam armazenados, transmitidos e processados por computadores.

Exemplo:
Morse Code: 1840



Caractere	Decimal	Binário
A	0	00000000
B	1	00000001
C	2	00000010
D	3	00000011
...
Z	25	00011001
a	26	00011010
b	27	00011010
...
z	51	00110011
0	52	00110100
1	53	00110101
...
!
,

Imagem: Rhey T. Snodgrass & Victor F. Camp, na Wikimedia Commons (https://commons.wikimedia.org/wiki/File:International_Morse_Code.svg)

ASCII: American Standard Code for Information Interchange

Original ASCII: 7 bits

0000000 (0)

1111111 (127)

É um encoding direto e de tamanho fixo:

- **Direto**: cada letra é mapeada diretamente para uma seqüência binária específica (A = 1000001)
- **Tamanho fixo**: cada letra binária tem 7 bits

0	<u>NUL</u>	16	<u>DLE</u>	32	<u>SP</u>	48	0	64	@	80	P	96	`	112	p
1	<u>SOH</u>	17	<u>DC1</u>	33	!	49	1	65	A	81	Q	97	a	113	q
2	<u>STX</u>	18	<u>DC2</u>	34	"	50	2	66	B	82	R	98	b	114	r
3	<u>ETX</u>	19	<u>DC3</u>	35	#	51	3	67	C	83	S	99	c	115	s
4	<u>EOT</u>	20	<u>DC4</u>	36	\$	52	4	68	D	84	T	100	d	116	t
5	<u>ENQ</u>	21	<u>NAK</u>	37	%	53	5	69	E	85	U	101	e	117	u
6	<u>ACK</u>	22	<u>SYN</u>	38	&	54	6	70	F	86	V	102	f	118	v
7	<u>BEL</u>	23	<u>ETB</u>	39	'	55	7	71	G	87	W	103	g	119	w
8	<u>BS</u>	24	<u>CAN</u>	40	(56	8	72	H	88	X	104	h	120	x
9	<u>HT</u>	25	<u>EM</u>	41)	57	9	73	I	89	Y	105	i	121	y
10	<u>LF</u>	26	<u>SUB</u>	42	*	58	:	74	J	90	Z	106	j	122	z
11	<u>VT</u>	27	<u>ESC</u>	43	+	59	;	75	K	91	[107	k	123	{
12	<u>FF</u>	28	<u>FS</u>	44	,	60	<	76	L	92	\	108	l	124	
13	<u>CR</u>	29	<u>GS</u>	45	-	61	=	77	M	93]	109	m	125	}
14	<u>SO</u>	30	<u>RS</u>	46	.	62	>	78	N	94	^	110	n	126	~
15	<u>SI</u>	31	<u>US</u>	47	/	63	?	79	O	95	_	111	o	127	<u>DEL</u>

ASCII: American Standard Code for Information Interchange

Extended ASCII: 8 bits

00000000 (0)

11111111 (255)

É um encoding direto e de tamanho fixo:

- **Direto:** cada letra é mapeada diretamente para uma seqüência binária específica (A = 01000001)
- **Tamanho fixo:** cada letra binária tem 8 bits

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0	NUL	SOH	STX	ETX	EOQ	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SS	SI
0x00	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
16	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x10	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
32	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0x20	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
48	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x30	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
64	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x40	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
80	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0x50	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
96	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x60	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
112	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
0x70	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
128	€		,	f	„	…	†	‡	^	%o	Š	‹	œ		Ž	
0x80	128		130	131	132	133	134	135	136	137	138	139	140		142	
144		'	'	“	”	•	—	—	~	™	š	›	œ		ž	ÿ
0x90		145	146	147	148	149	150	151	152	153	154	155	156		158	159
160		i	ç	£	¤	¥	¦	§	¨	©	ª	«	¬	173	®	¯
0xA0	160	161	162	163	164	165	166	167	168	169	170	171	172		174	175
176	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼	½	¾	¿
0xB0	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
192	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
0xC0	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
208	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
0xD0	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
224	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
0xE0	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
240	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ
0xF0	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

ASCII Code Chart (<https://www.ascii-code.com/codechart>)

Representação de texto com ASCII

0	NUL	16	DLE	32	SP	48	0	64	@	80	P	96	`	112	p
1	SOH	17	DC1	33	!	49	1	65	A	81	Q	97	a	113	q
2	STX	18	DC2	34	"	50	2	66	B	82	R	98	b	114	r
3	ETX	19	DC3	35	#	51	3	67	C	83	S	99	c	115	s
4	EOT	20	DC4	36	\$	52	4	68	D	84	T	100	d	116	t
5	ENQ	21	NAK	37	%	53	5	69	E	85	U	101	e	117	u
6	ACK	22	SYN	38	&	54	6	70	F	86	V	102	f	118	v
7	BEL	23	ETB	39	'	55	7	71	G	87	W	103	g	119	w
8	BS	24	CAN	40	(56	8	72	H	88	X	104	h	120	x
9	HT	25	EM	41)	57	9	73	I	89	Y	105	i	121	y
10	LF	26	SUB	42	*	58	:	74	J	90	Z	106	j	122	z
11	VT	27	ESC	43	+	59	;	75	K	91	[107	k	123	{
12	FF	28	FS	44	,	60	<	76	L	92	\	108	l	124	
13	CR	29	GS	45	-	61	=	77	M	93]	109	m	125	}
14	SO	30	RS	46	.	62	>	78	N	94	^	110	n	126	~
15	SI	31	US	47	/	63	?	79	O	95	_	111	o	127	DEL

Oi!

79

105

33

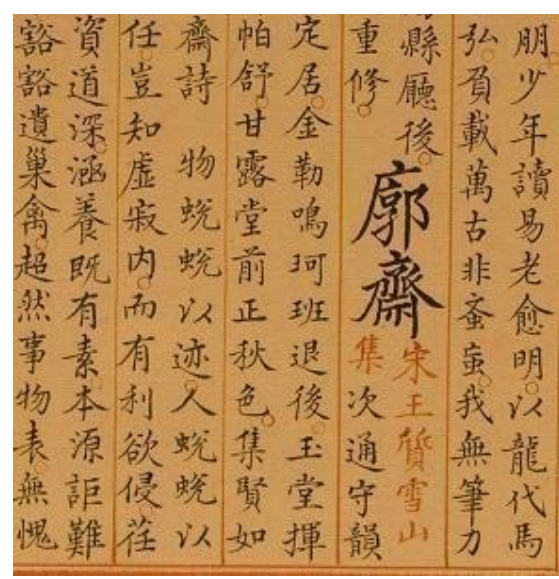
01001111

01101001

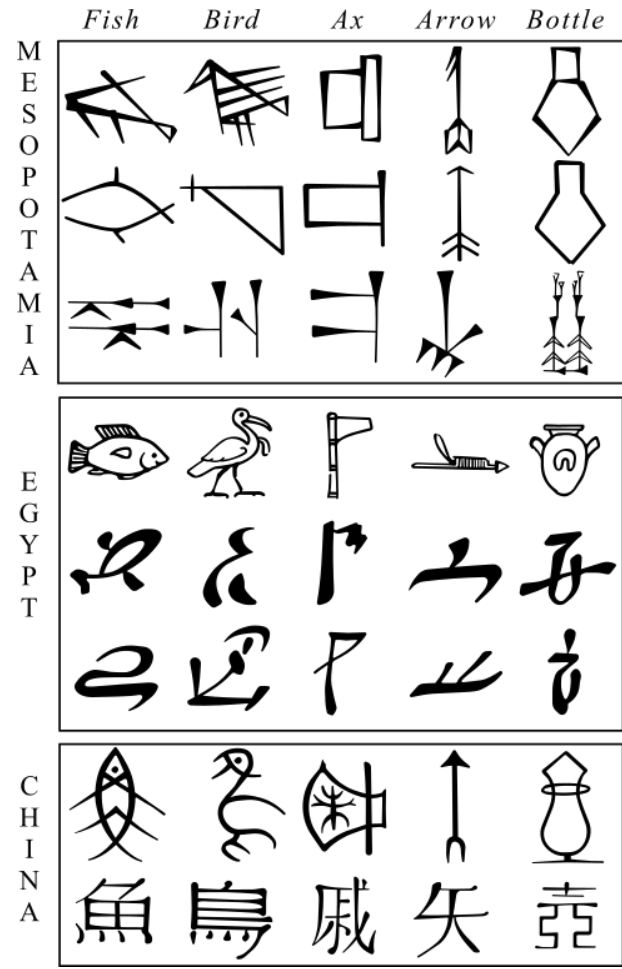
00100001

Limitações do ASCII

- Foco no sistema de escrita latino
- **Maioria dos sistemas de escrita não é representado**



Xie Jin, na Wikipedia
 (https://en.wikipedia.org/wiki/File:Kuozhai.jpg)



Gaston Maspero, na Wikipedia
 (https://en.wikipedia.org/wiki/File:Comparative_evolution_of_Cuneiform,_Egyptian_and_Chinese_characters.svg)

	4	3	2	1	2	3	4	
CORRESPONDING ENGLISH	ARCHAIC ROMAN	ARCHAIC GREEK	PHENICIAN	BRĀHMA	DEVELOPMENTS OF BRĀHMA			MODERN NĀGARI
A	Α	Α	𐤀	𑀅	𑀇	𑀈	𑀉	अ
K	Κ	𐀀	𐤊	𑀇	𑀇	𑀈	𑀉	क
G	Γ	𐀁	𐤋	𑀇	𑀇	𑀈	𑀉	ग
T	Τ	𐀂	𐤌	𑀇	𑀇	𑀈	𑀉	त
TH	⊙	⊙	⊙	⊙	⊙	𑀇	𑀈	थ
D	Δ	Δ	𑀇	𑀇	𑀇	𑀈	𑀉	द
P	ρ	ρ	𑀇	𑀇	𑀇	𑀈	𑀉	प
B	β	β	𑀇	𑀇	𑀇	𑀈	𑀉	ब
Y	Υ	Υ	𑀇	𑀇	𑀇	𑀈	𑀉	य
V	ν	ν	𑀇	𑀇	𑀇	𑀈	𑀉	व

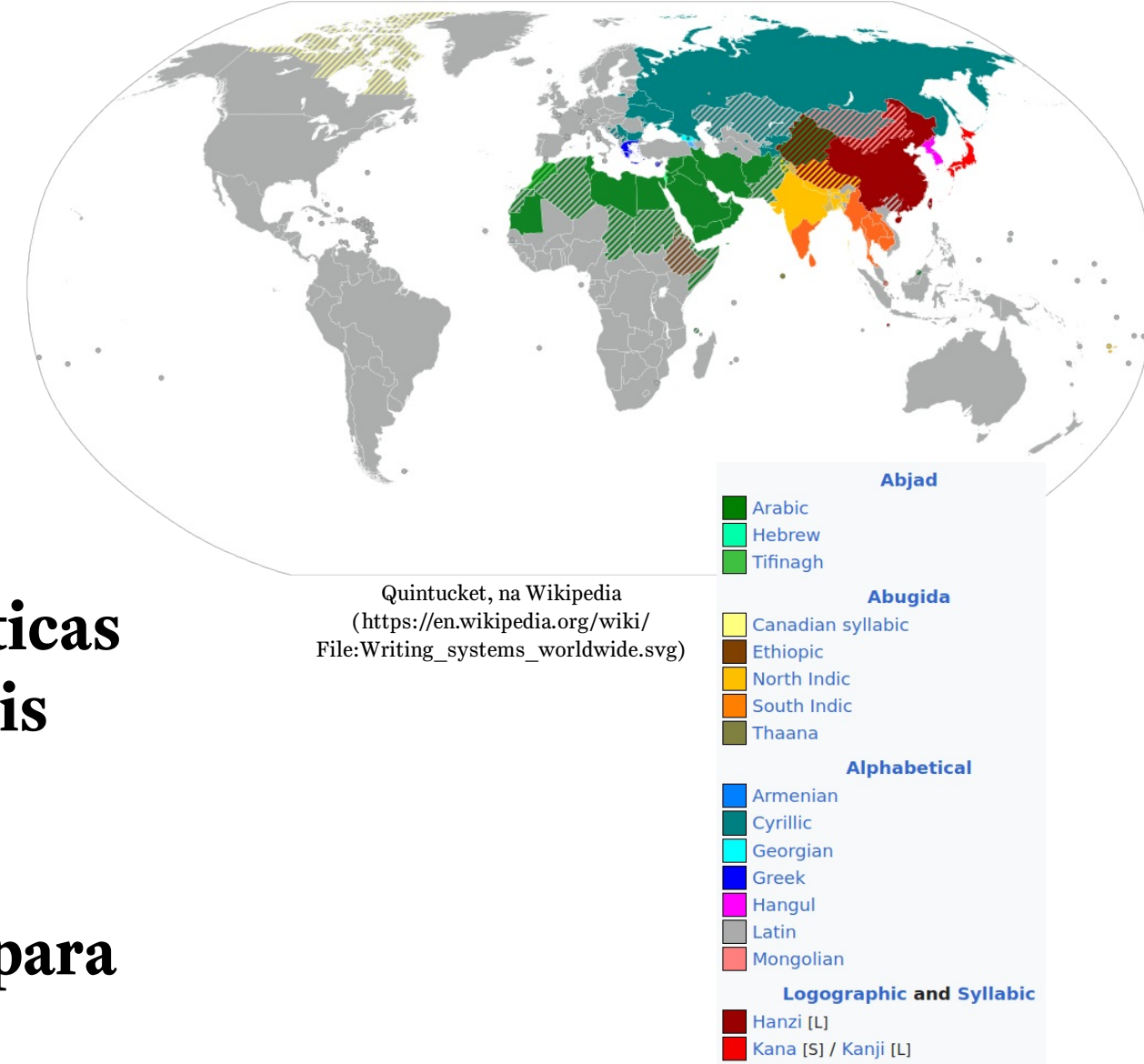
Monnier Williams, na Wikipedia
 (https://en.wikipedia.org/wiki/File:Sanskrit_Brahma_English_alphabets.jpg)

Limitações do ASCII

Sistema de escrita: método de representar visualmente a comunicação verbal através de um script e um conjunto de regras. Principais:

- **Alfabetos:** as letras representam os sons
- **Silábicos:** os símbolos representam as sílabas
- **Logográficos:** os símbolos representam unidades semânticas
- **Abjads:** semelhante aos alfabetos, não representa as vogais
- **Abugidas:** símbolos representam par cosoante-vogal

Script: coleção de letras e outros sinais escritos, utilizados para representar informação textual em um ou mais sistemas de escrita. Alguns scripts suportam apenas um único sistema de escrita e linguagem (script armênico), e outros suportam muitos sistemas de escrita e linguagens (script latino: inglês, francês, alemão, italiano, português, ...)



Unicode

Criado em 1991 para representar todos os scripts de todos os sistemas de escrita (em uso ou extintos). Atualmente na versão 15.1 (setembro/2023), contendo:

- 161 scripts
- 149.813 caracteres

Não é diretamente um encoding, como o ASCII. O Unicode, na verdade, é um grande "catálogo" de letras, símbolos, caracteres, etc., cada uma com um **código numérico único, exclusivo e universal**, chamado de "Unicode **Code Point**".

Os code points estabelecidos vão do número **0** até **1.114.111** (1.114.112 números), dos quais a faixa de números **55.296** até **57.343** (2.048 números) é reservada, sobrando então o total de **1.112.064 code points** disponíveis para uso.



<https://home.unicode.org>

Unicode: sobre os code points

A faixa numérica de code points (0 até 1.114.111) é representada em hexadecimal, com o prefixo "U+", seguidos de pelo menos quatro dígitos. A faixa de code points é então:

U+0000	0
U+10FFFF	1.114.111

Existem sites que permitem consultar os code points e caracteres Unicode, tais como:

- Codepoints (<https://codepoints.net>)
- Unicode Explorer (<https://unicode-explorer.com>)

Lembre-se: os code points não são um encoding direto, ou seja, não correspondem diretamente a nenhum padrão de bits para representar os caracteres (veremos em breve como o encoding é feito).

Unicode: exemplos de code points

ASCII punctuation and symbols

U+0020	U+0021	U+0022	U+0023	U+0024	U+0025	U+0026
'	()	*			





























































Extended Arabic letters

U+0679	U+067A	U+067B	U+067C	U+067D	U+067E	U+067F
U+0680	U+0681	U+0682	U+0683	U+0684	U+0685	U+0686
U+0687	U+0688	U+0689	U+068A	U+068B	U+068C	U+068D

Lowercase Latin alphabet

U+0061	U+0062	U+0063	U+0064	U+0065	U+0066	U+0067
U+0068	U+0069	U+006A	U+006B	U+006C	U+006D	U+006E
U+006F	U+0070	U+0071	U+0072	U+0073	U+0074	U+0075
U+0076	U+0077	U+0078	U+0079	U+007A		

Unicode: exemplos de code points

U+1F600  Grinning Face	U+1F601  Grinning Face with Smiling Eyes	U+1F602  Face with Tears of Joy	U+1F603  Smiling Face with Open Mouth	U+1F604  Smiling Face with Open Mouth and Smiling Eyes	U+1F605  Smiling Face with Open Mouth and Cold Sweat	U+1F606  Smiling Face with Open Mouth and Tightly-Closed Eyes	U+1F607  Smiling Face with Halo	U+1F608  Smiling Face with Horns	U+1F609  Winking Face	U+1F60A  Smiling Face with Smiling Eyes	U+1F60B  Face Savouring Delicious Food	U+1F60C  Relieved Face	U+1F60D  Smiling Face with Heart-Shaped Eyes	U+1F60E  Smiling Face with Sunglasses
U+1F60F  Smirking Face	U+1F610  Neutral Face	U+1F611  Expressionless Face	U+1F612  Unamused Face	U+1F613  Face with Cold Sweat	U+1F614  Pensive Face	U+1F615  Confused Face	U+1F616  Confounded Face	U+1F617  Kissing Face	U+1F618  Face Throwing A Kiss	U+1F619  Kissing Face with Smiling Eyes	U+1F61A  Kissing Face with Closed Eyes	U+1F61B  Face with Stuck-Out Tongue	U+1F61C  Face with Stuck-Out Tongue and Winking Eye	U+1F61D  Face with Stuck-Out Tongue and Tightly-Closed Eyes
U+1F61E  Disappointed Face	U+1F61F  Worried Face	U+1F620  Angry Face	U+1F621  Pouting Face	U+1F622  Crying Face	U+1F623  Persevering Face	U+1F624  Face with Look of Triumph	U+1F625  Disappointed But Relieved Face	U+1F626  Frowning Face with Open Mouth	U+1F627  Anguished Face	U+1F628  Fearful Face	U+1F629  Weary Face	U+1F62A  Sleepy Face	U+1F62B  Tired Face	U+1F62C  Grimacing Face
U+1F62D  Loudly Crying Face	U+1F62E  Face with Open Mouth	U+1F62F  Hushed Face	U+1F630  Face with Open Mouth and Cold Sweat	U+1F631  Face Screaming In Fear	U+1F632  Astonished Face	U+1F633  Flushed Face	U+1F634  Sleeping Face	U+1F635  Dizzy Face	U+1F636  Face Without Mouth	U+1F637  Face with Medical Mask	U+1F638  Grinning Cat Face with Smiling Eyes	U+1F639  Cat Face with Tears of Joy	U+1F63A  Smiling Cat Face with Open Mouth	U+1F63B  Smiling Cat Face with Heart-Shaped Eyes

Representação de texto com Unicode



Imagem: CS50 Lecture 0 (<https://cdn.cs50.net/2022/fall/lectures/0/lecture0.pdf>)

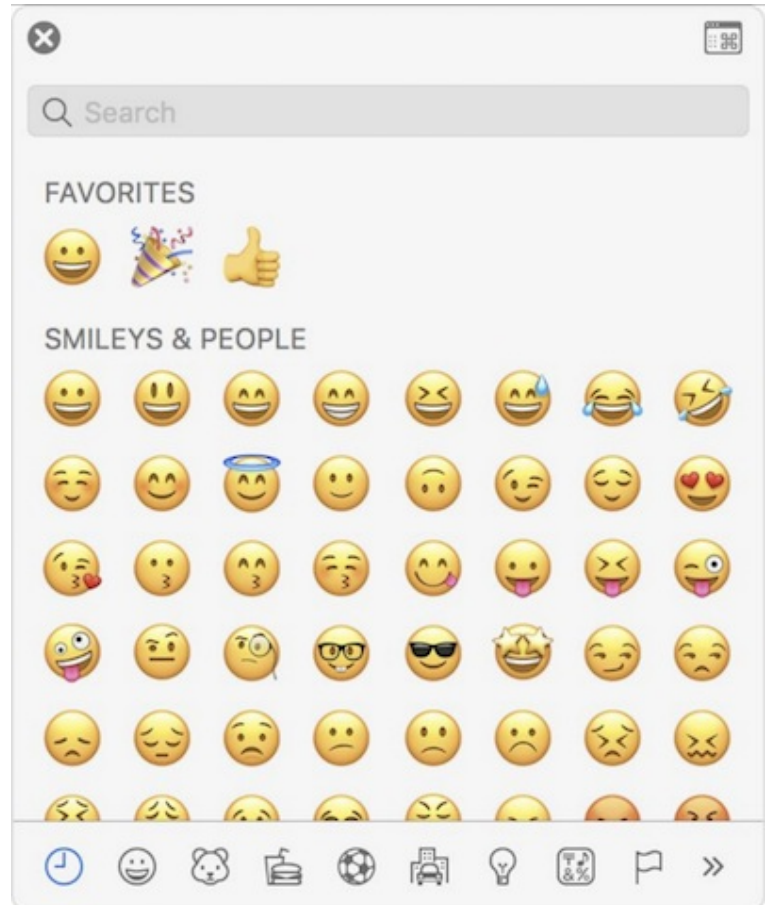


Imagem: CS50 Lecture 0 (<https://cdn.cs50.net/2022/fall/lectures/0/lecture0.pdf>)



1F602
128514
11111011000000010

hexadecimal
decimal
binário

Encoding dos code points

Os code points do Unicode não representam um mapeamento direto em binário. Para esse mapeamento o Unicode utiliza diversos esquemas de encoding. Os principais são:

- Unicode Transformation Format (UTF)
- Universal Coded Character Set (UCS)

UTF-8: tamanho variável: utiliza de **1 a 4 blocos de 8 bits** para cada code point; compatível com ASCII (7 bits).

UTF-16: tamanho variável: utiliza de **1 a 2 blocos de 16 bits** para cada code point; não compatível com ASCII (7 bits).

UTF-32: tamanho fixo: utiliza **1 bloco de 32 bits** para cada code point; não compatível com ASCII (7 bits).

Outros: UTF-EBCDIC, UCS-2, UCS-4, ...

Mais usado hoje em dia: UTF-8.

Encoding dos code points

Imagens: Codepoints (<https://codepoints.net>)

UTFs: Online Tools (<https://onlinetools.com/unicode/convert-unicode-to-binary>)



U+0041 (65)
UTF-8 : 01000001
UTF-16: 0000000001000001
UTF-32: 0000000000000000000000000000000001000001



U+00A9 (169)
UTF-8 : 11000010 10101001
UTF-16: 0000000010101001
UTF-32: 0000000000000000000000000000000010101001



U+2622 (9762)
UTF-8 : 11100010 10011000 10100010
UTF-16: 0010011000100010
UTF-32: 00000000000000000000000010011000100010



U+20046 (131142)
UTF-8 : 11110000 10100000 10000001 10000110
UTF-16: 1101100001000000 1101110001000110
UTF-32: 00000000000000001000000000001000110

Unicode: não define a apresentação final

Smileys & Emotion														
face-smiling														
Nº	Code	Browser	Appl	Goog	FB	Wind	Twtr	Joy	Sams	GMail	SB	DCM	KDDI	CLDR Short Name
1	U+1F600										—	—	—	grinning face
2	U+1F603													grinning face with big eyes
3	U+1F604											—	—	grinning face with smiling eyes
4	U+1F601													beaming face with smiling eyes
5	U+1F606										—		—	grinning squinting face
6	U+1F605										—		—	grinning face with sweat
7	U+1F923									—	—	—	—	rolling on the floor laughing
8	U+1F602											—		face with tears of joy
9	U+1F642										—	—	—	slightly smiling face
10	U+1F643									—	—	—	—	upside-down face

Unicode Emoji List (<https://unicode.org/emoji/charts-15.0/full-emoji-list.html>)

Roboto Mono Variable (2 axes) | Christian Robertson

Computação

Oswald Variable (1 axis) | Vernon Adams, Kalapi Gajjar, Cyreal

Computação

Noto Sans Variable (3 axes) | Google

Computação

Marhey Variable (1 axis) | Nur Syamsi, Bustanul Arifin




























































Computação

Raleway Variable (2 axes) | Matt McInerney, Pablo Impallari, Rodrigo Fuenzalida

Computação

Google Fonts (<https://fonts.google.com>)

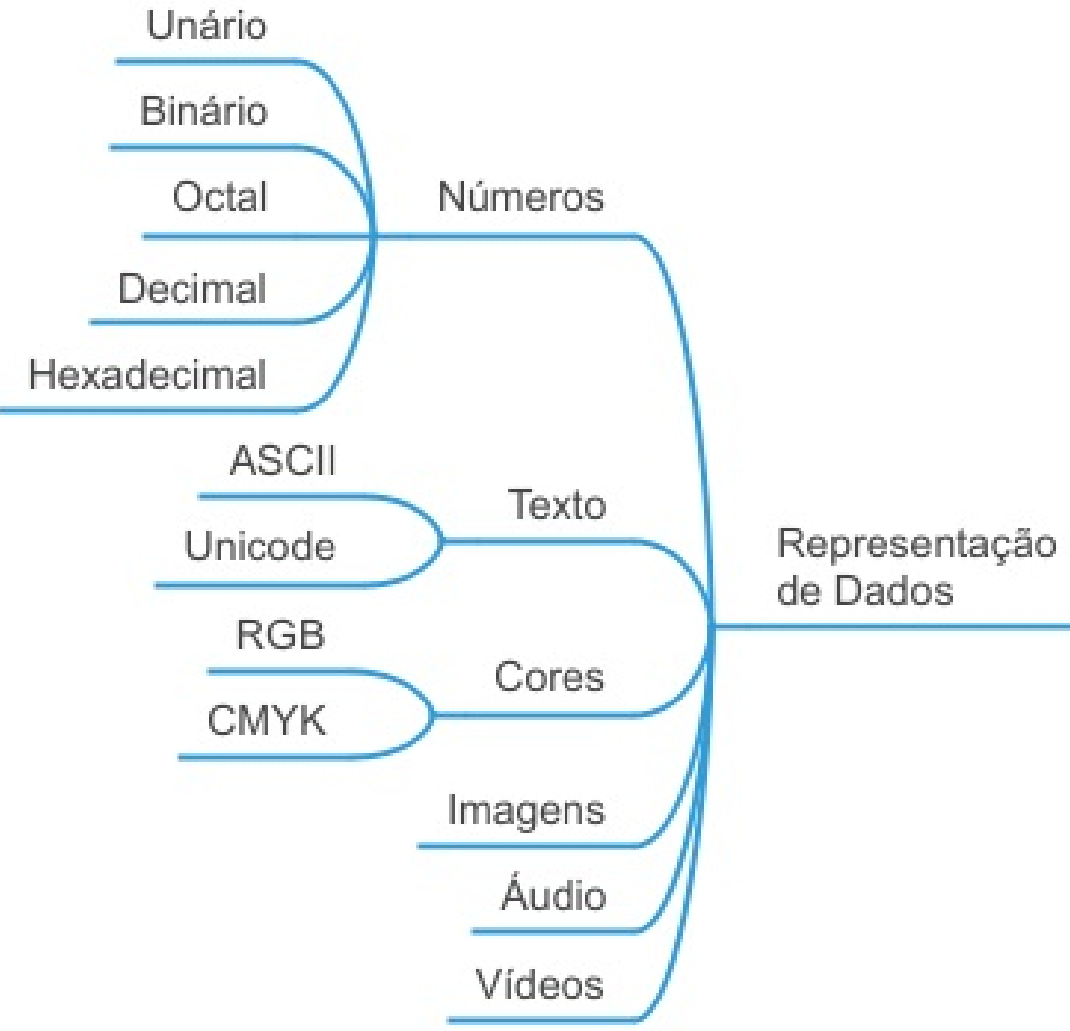
Unicode: mais de um code point pode ser utilizado

316	U+1F468 U+1F3FB U+200D U+1F980									—	—	—	—	man: light skin tone, red hair
301	U+1F9D4 U+1F3FB									—	—	—	—	person: light skin tone, beard
302	U+1F9D4 U+1F3FC									—	—	—	—	person: medium-light skin tone, beard
303	U+1F9D4 U+1F3FD									—	—	—	—	person: medium skin tone, beard
304	U+1F9D4 U+1F3FE									—	—	—	—	person: medium-dark skin tone, beard
305	U+1F9D4 U+1F3FF									—	—	—	—	person: dark skin tone, beard
139	U+2764 U+FE0F U+200D U+1F525				—	—	—	—	—	—	—	—	—	heart on fire
273	U+1F64B U+200D U+2640 U+FE0F									—	—	—	—	woman raising hand

Unicode Emoji List (<https://unicode.org/emoji/charts-15.0/full-emoji-list.html>)

Unicode Emoji Modifier Sequences (<https://unicode.org/emoji/charts-15.0/full-emoji-modifiers.html>)

Representação de dados: cores



Monitor

Impressão

Representação de cor: RGB (red-green-blue)

Para representar as cores em um monitor, padronizou-se que a cor de cada pixel da tela é determinada por 24 bits (3 Bytes):

- R: a quantidade de vermelho (de 0 a 255)
- G: a quantidade de verde (de 0 a 255)
- B: a quantidade de azul (de 0 a 255)

É um modelo **aditivo**, onde raios de luz de diferentes cores são sobrepostos (somados) para dar o resultado final.

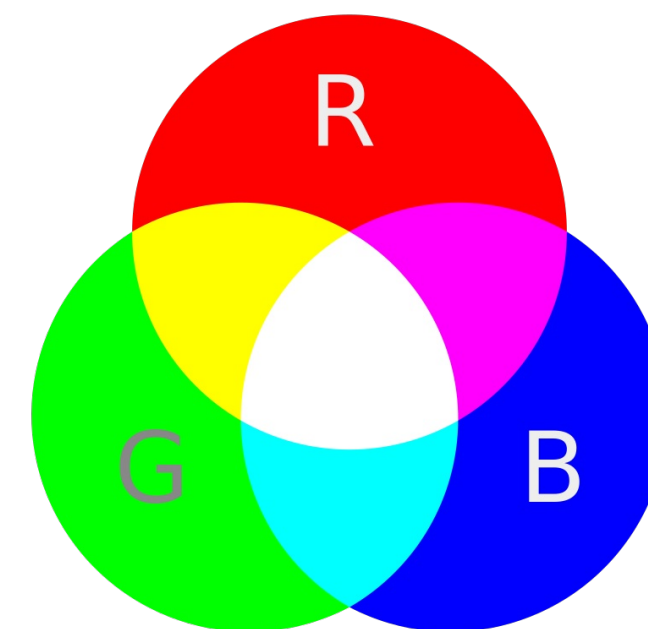
- 0: ausência da cor
- 255: máximo da cor

RGB (0, 0, 0) = preto

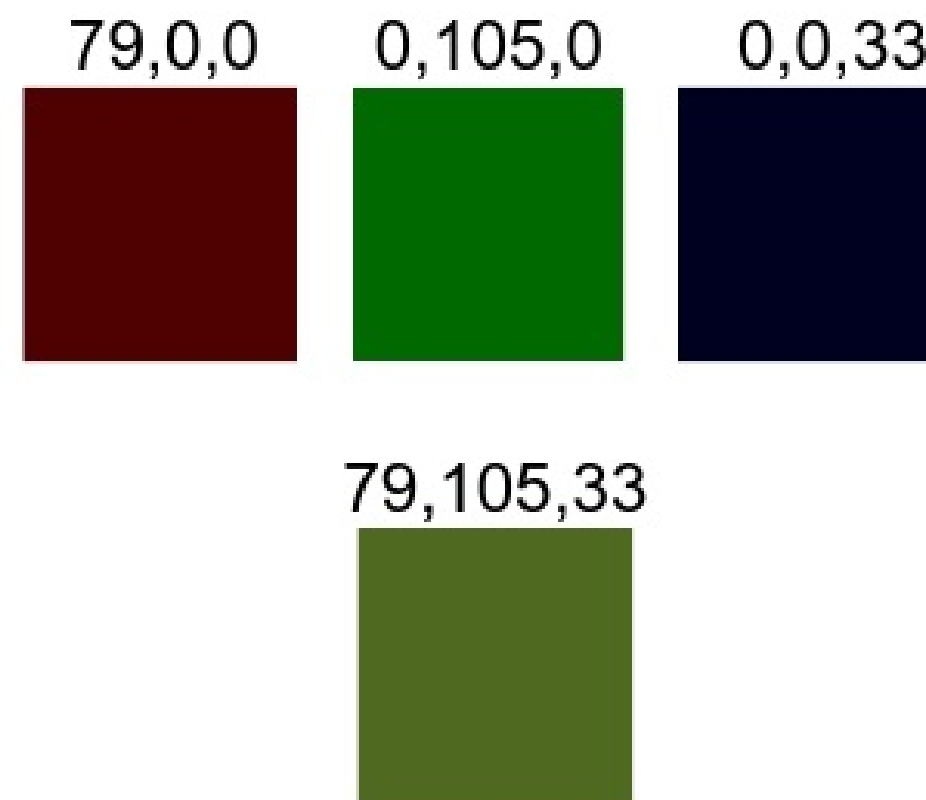
RGB (255, 255, 255) = branco

RGB (127, 127, 127) = cinza

Total de cores: 16.777.216



Immanuelle, na Wikipedia
(https://en.wikipedia.org/wiki/File:Venn_diagram_rgb.svg)



Representação de cor: RGB (red-green-blue)

RGB Calculator



`rgb(255, 215, 65)`

`#ffd741`

`hsl(47, 100%, 63%)`

R:  255

G:  215

B:  65

[Use this color in our Color Picker](#)

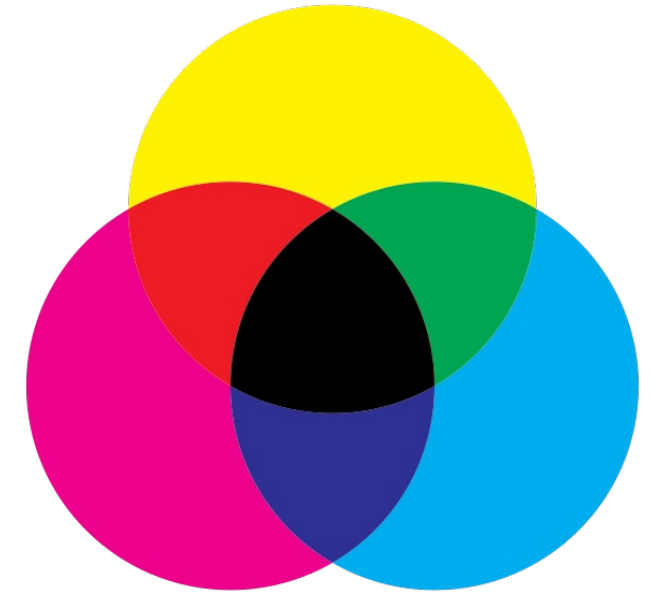
Representação de cor: CMYK (cyan-magenta-yellow-black)

Para representar as cores impressas, padronizou-se que a cor de cada ponto no papel é determinada por 4 cores:

- Ciano
- Magenta
- Amarelo
- Preto

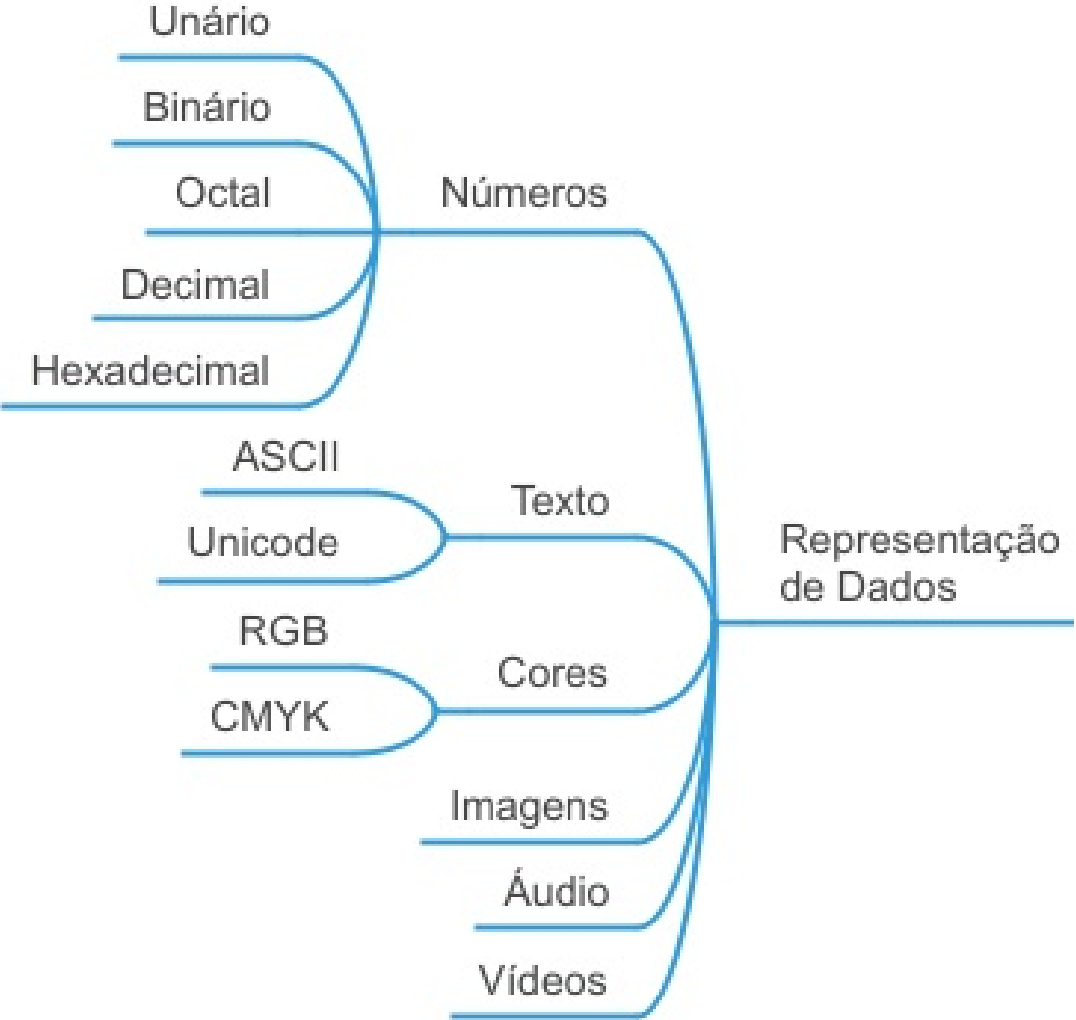
É um modelo **subtrativo** pois as tintas "subtraem" as cores vermelha, verde e azul, da luz branca que seria refletida pelo papel.

- Luz branca menos vermelho = ciano
- Luz branca menos verde = magenta
- Luz branca menos azul = amarelo



Youssef Abdelhamed, na Wikipedia
(https://en.wikipedia.org/wiki/File:CMYK_color_model.svg)

Representação de dados: imagens



Fotos

Resolução, profundidade e tamanho

Compressão
- **com perda**
- **sem perda**

Padrões:
- **BMP**
- **JPEG**
- ...

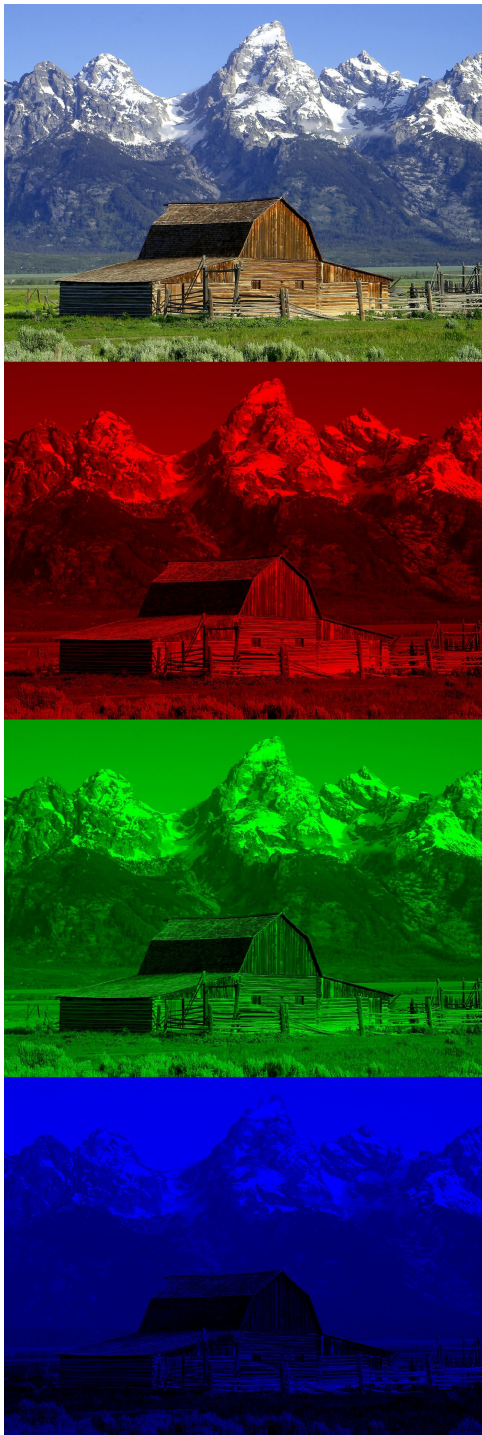
"Enhance!"

Como representar uma fotografia?



Jon Sullivan, na Wikipedia
(https://en.wikipedia.org/wiki/File:Barns_grand_tetons.jpg)

Como representar uma fotografia?



Mike1024, na Wikipedia
(https://en.wikipedia.org/wiki/File:Barn_grand_tetons_rgb_separation.jpg)

Armazenando a quantidade de vermelho, verde e azul (RGB) de cada pixel da imagem.

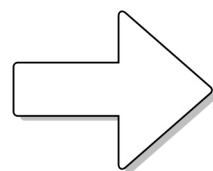
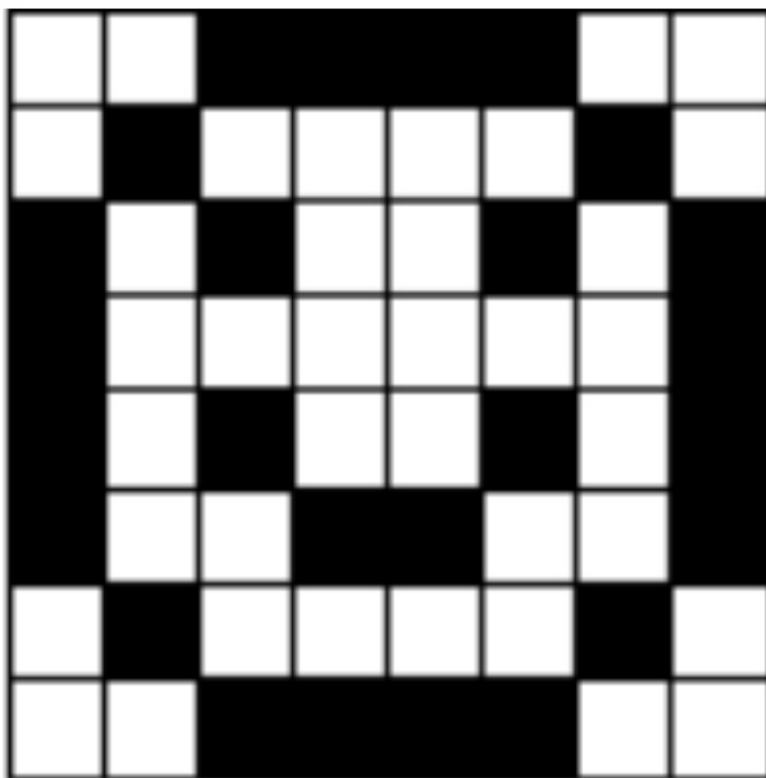


Imagem: CS50 Lecture 0 (<https://cdn.cs50.net/2022/fall/lectures/0/lecture0.pdf>)

Como representar uma fotografia? um exemplo simples

Todas as imagens são retangulares, por natureza. Alguns pixels podem ser transparentes para dar a impressão de curvas mas, por fim, as imagens são retangulares.

A imagem abaixo, de 8x8 pixels, representa branco ou preto com apenas 1 bit. Uma maneira de representar essa imagem é um "mapa de bits", um **bitmap**:

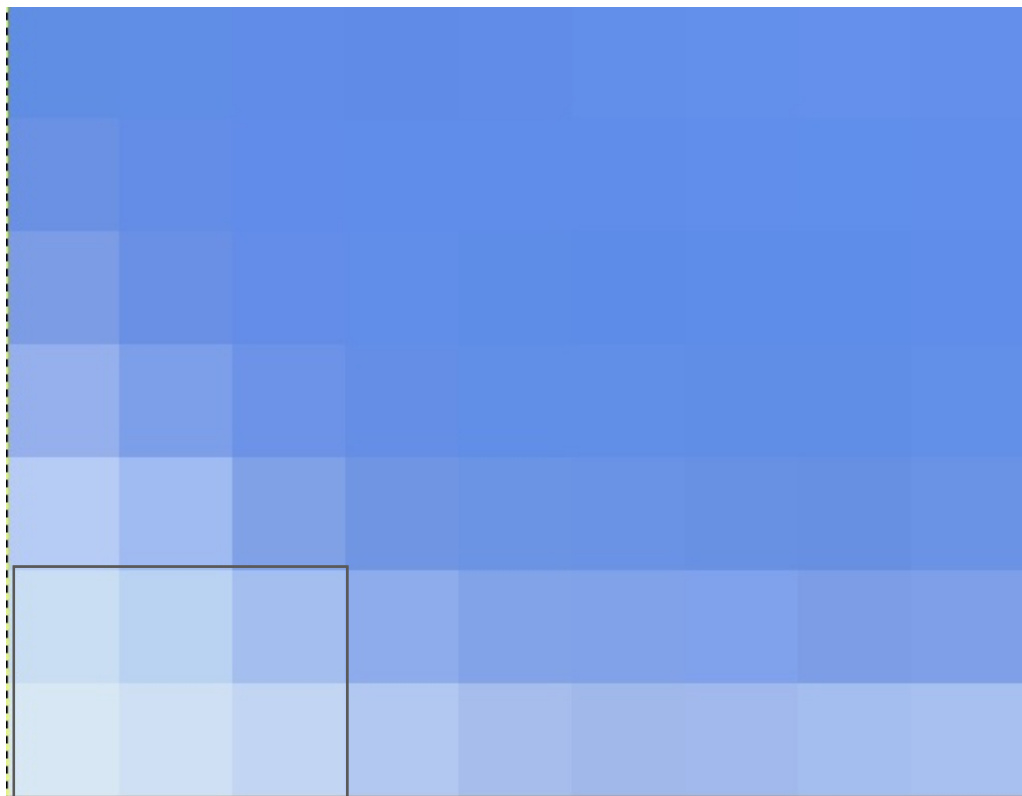


```
11000011
10111101
01011010
01111110
01011010
01100110
10111101
11000011
```

Imagem: CS50 Technology Lecture 3
(<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)



Como representar uma fotografia? bitmaps



(78, 86, 95) (73, 82, 95) (64, 74, 93)
(84, 90, 95) (81, 87, 96) (76, 83, 95)

01001110,01010110,01011111 01001001,01010010,01011111 01000000,01001010,01011101
01010100,01011010,01011111 01010001,01010111,01100000 01001100,01010011,01011111

01001110010101100101111101001001010100100101111101000000100101001011101
010101000101101001011111010100010101011101100000010011000101001101011111

Microsoft, na Wikipedia ([https://en.wikipedia.org/wiki/File:Bliss_\(Windows_XP\).png](https://en.wikipedia.org/wiki/File:Bliss_(Windows_XP).png))

Imagem: resolução, profundidade e tamanho



Jon Sullivan, na Wikipedia

(https://en.wikipedia.org/wiki/File:Barns_grand_tetons.jpg)

Resolução: quantidade de pixels horizontais e verticais que a imagem possui.

Profundidade de cor: quantidade de bits utilizados para representar cada cor.

Foto: 1024 x 765 pixels
783.360 pixels no total

Tamanho: 783.360 pixels
24 bits por pixel
 $783.360 \times 24 = 18.800.640$ bits
= 2.350.080 Bytes
= 2.295 KiB
= 2,2 MiB

Imagem: compressão

Calculamos 2.295 KiB, mas a imagem tem só 303 KiB. Por quê?

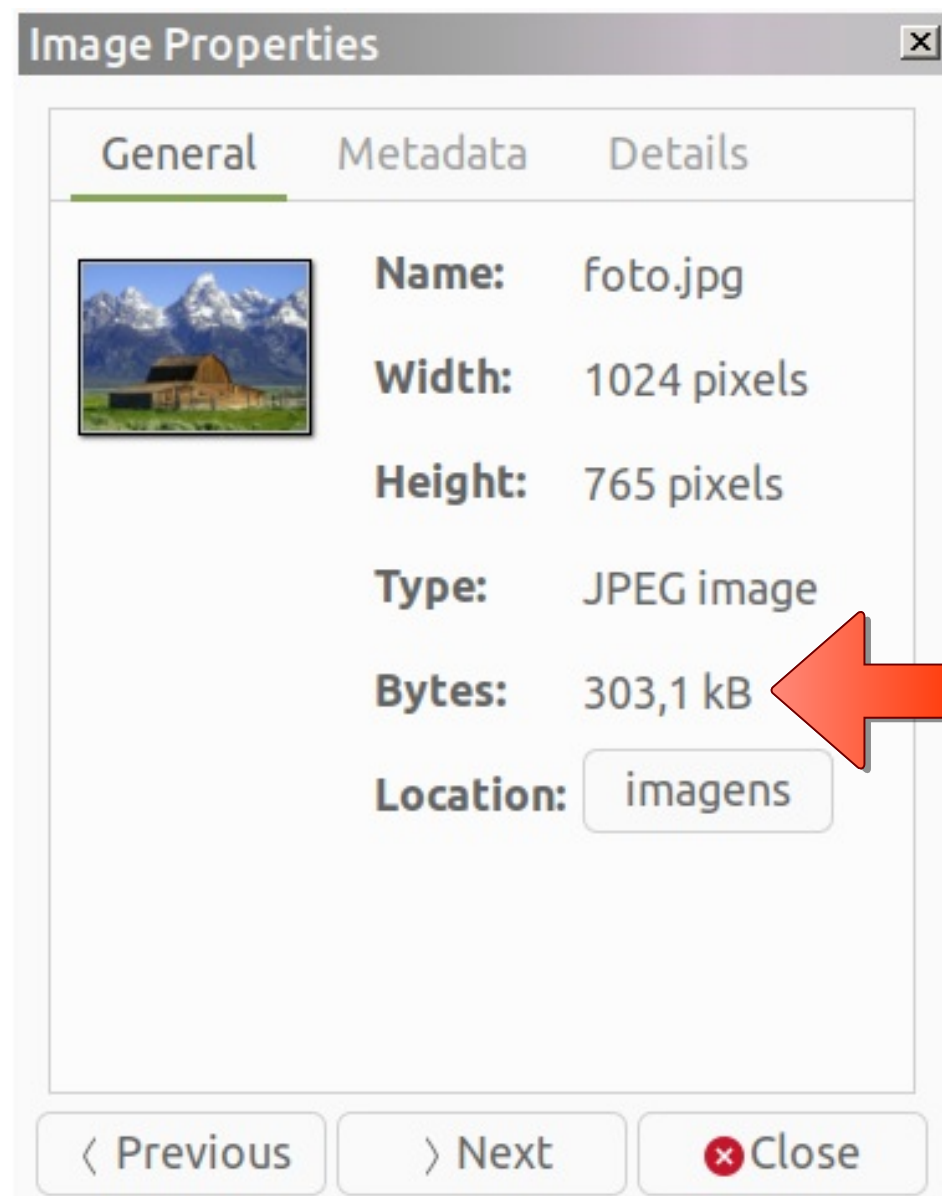


Imagem: compressão

Método para diminuir o tamanho do arquivo final. Pode ser feita basicamente de 2 modos:

- Com perda
- Sem perda

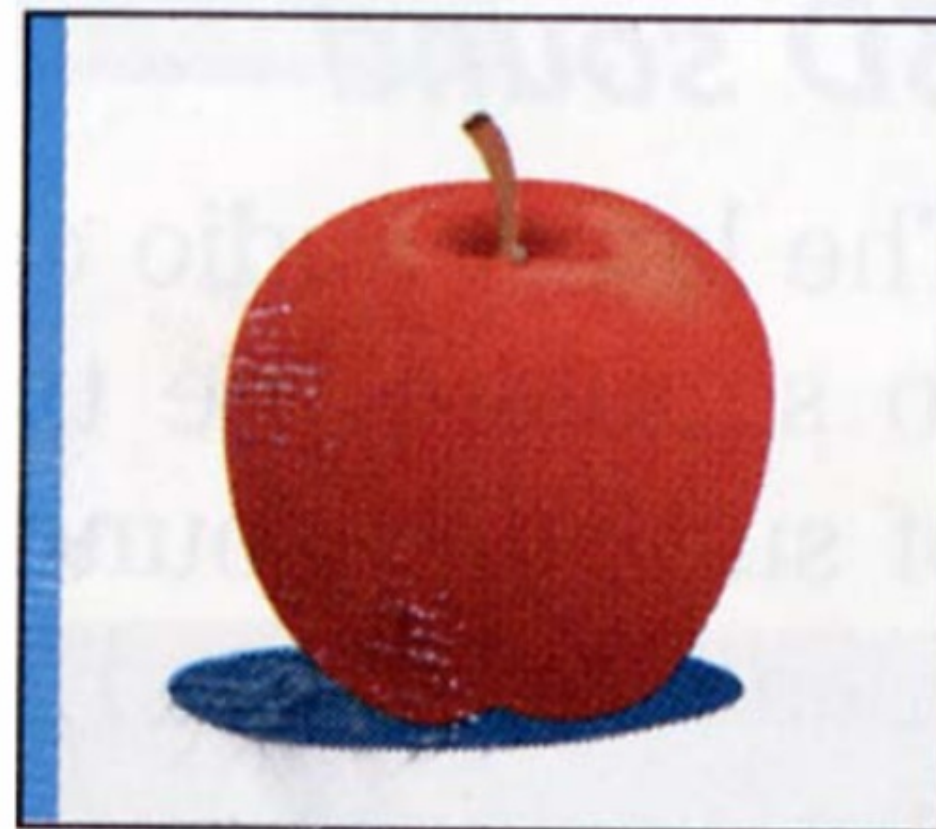
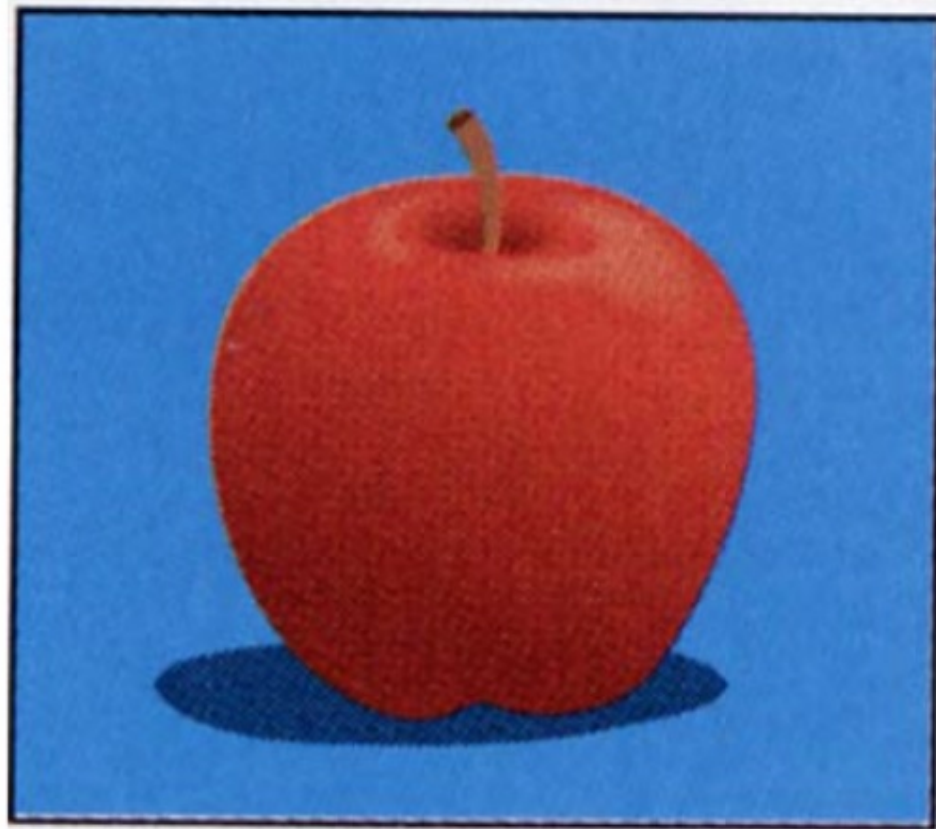


Imagem: compressão

Método para diminuir o tamanho do arquivo final. Pode ser feita basicamente de 2 modos:

- Com perda
- Sem perda



Imagem: CS50 Technology Lecture 3
(<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

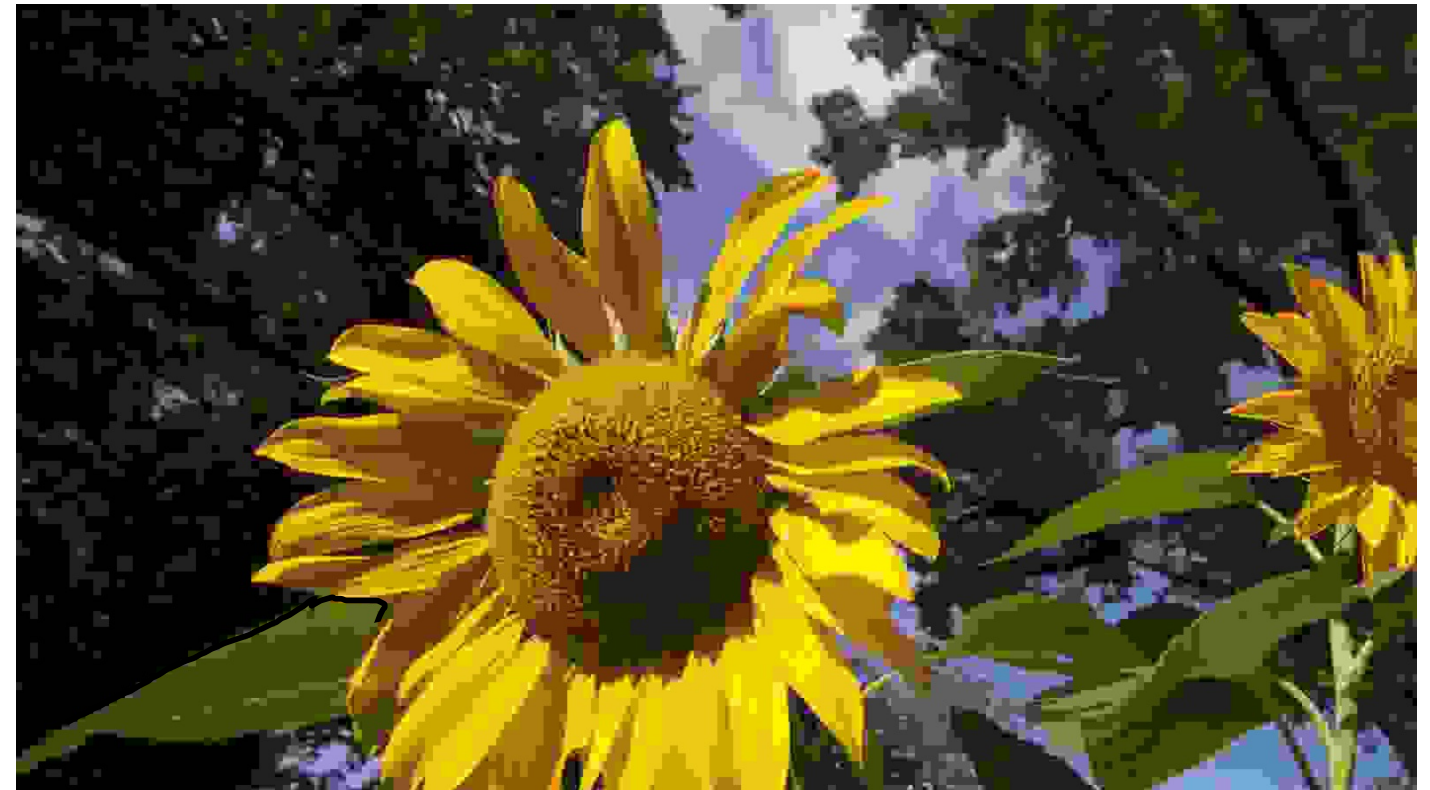


Imagem: CS50 Technology Lecture 3
(<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

Imagem: formatos de arquivos

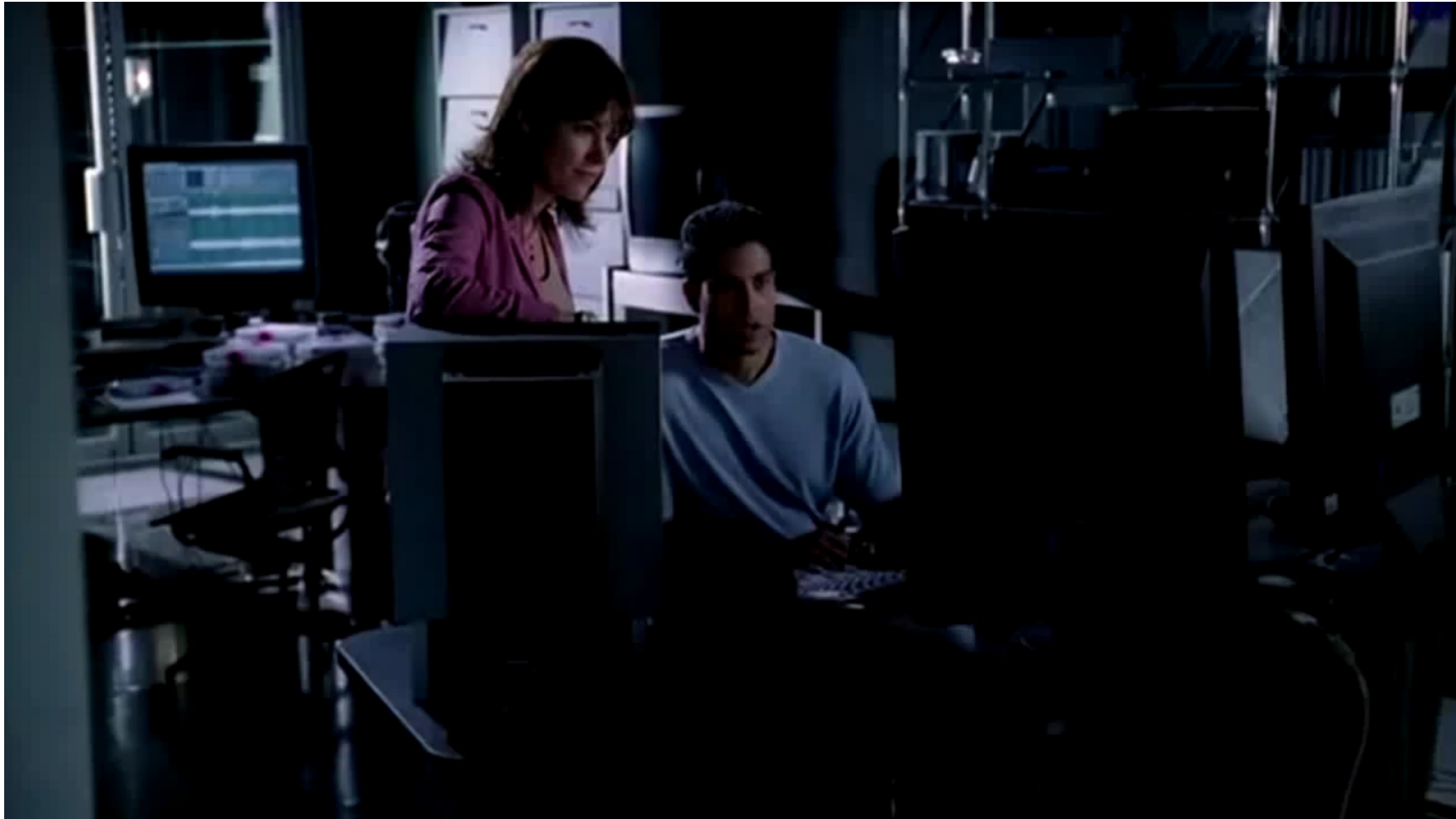
Existem diferentes **formatos de arquivos** de imagem, cada um armazena os bits de forma diferente, alguns com compressão (com ou sem perda) e outros sem compressão.

Formato de arquivo: uma maneira de armazenar os bits de um arquivo no disco, de modo que certos softwares saibam como interpretar esses padrões de 0s e 1s.

Os principais formatos de arquivos para imagens são:

- **BMP:** bitmap
- **GIF:** para imagens com baixa qualidade (8 bits): icons, memes (gifs animados)
- **JPEG:** para imagens com boa qualidade (24 bits), com perda
- **PNG:** para imagens com alta qualidade (24 bits): múltiplos usos
- ...

"Enhance": é possível melhorar, aprimorar, uma imagem?



CSI Miami, Temporada 1, Episódio 9

Imagem: CS50 Technology Lecture 3 (<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

"Enhance": é possível melhorar, aprimorar, uma imagem?



CSI Miami, Temporada 1, Episódio 9

Imagem: CS50 Technology Lecture 3 (<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

"Enhance": é possível melhorar, aprimorar, uma imagem?



CSI Miami, Temporada 1, Episódio 9

Imagem: CS50 Technology Lecture 3 (<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

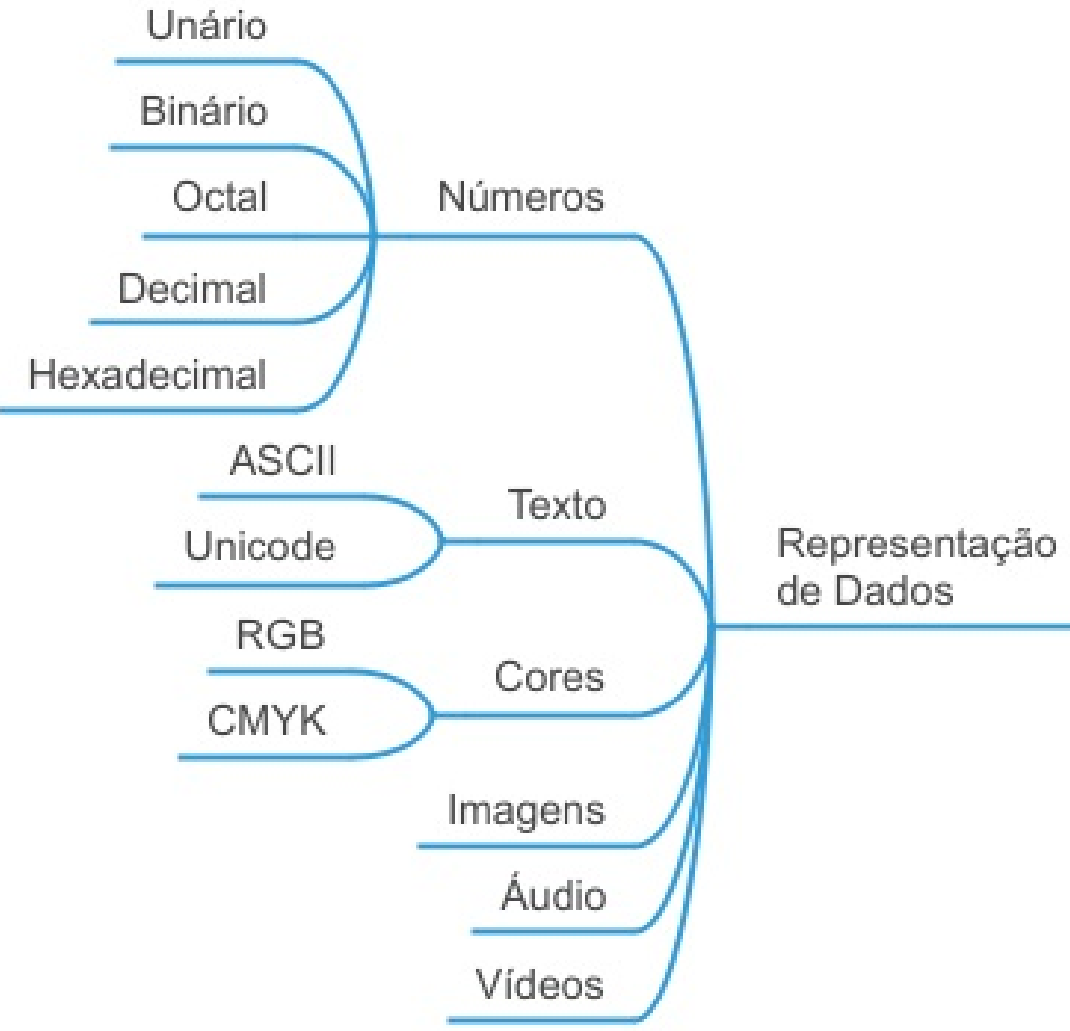
"Enhance": é possível melhorar, aprimorar, uma imagem?



CSI Miami, Temporada 1, Episódio 9

Imagem: CS50 Technology Lecture 3 (<https://cdn.cs50.net/cscie1a/2017/fall/lectures/multimedia/multimedia.pdf>)

Representação de dados: áudio



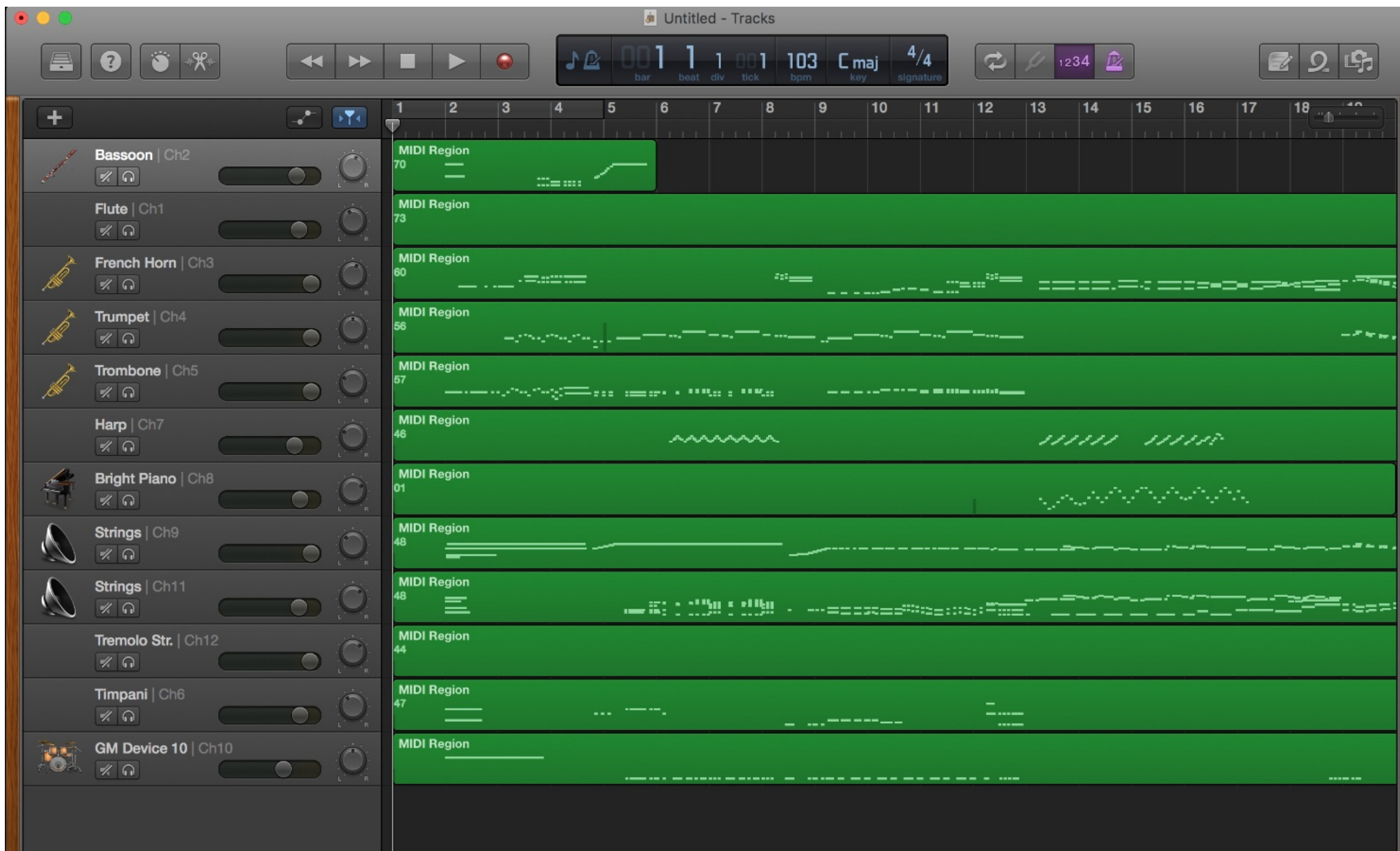
Formatos de áudio

Representação do som

Qualidade de áudio

Formatos de arquivos de áudio: MIDI (sintetização)

Musical Instrument Digital Interface (MIDI)



Armazena as notas dos instrumentos (um ou vários).

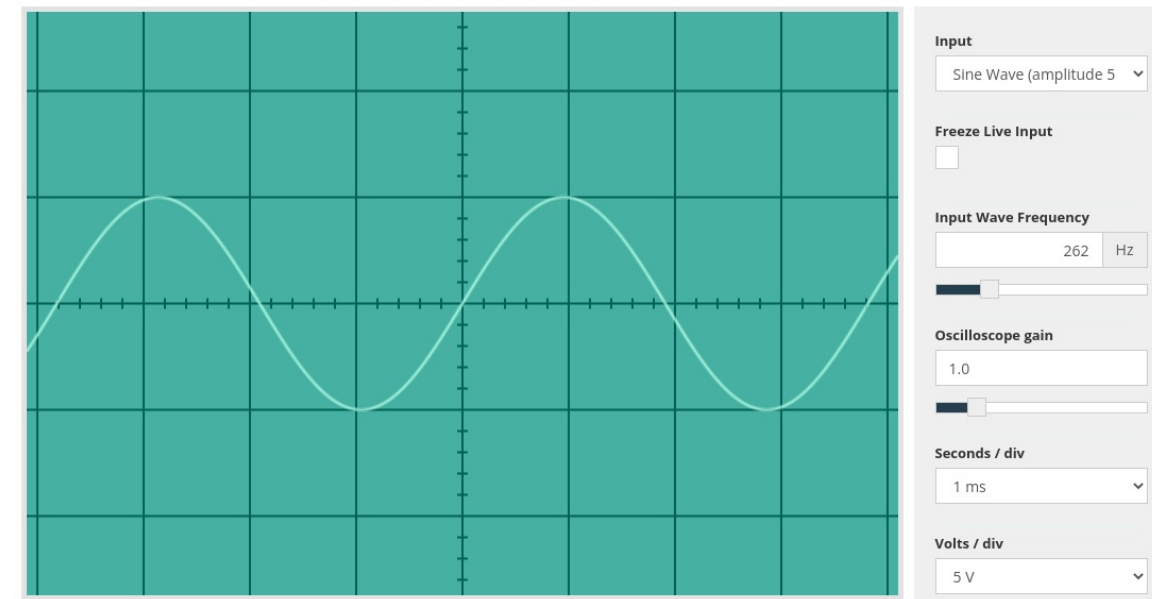
O computador sintetiza o som, não é uma gravação.

Outros formatos de áudio (gravação)

- **WAV: Waveform Audio File:** armazena dados não comprimidos, em alta qualidade
- **MP3: Motion Picture Experts Group Audio Level 3 Encoding**
áudio com compressão, tamanho reduzido, qualidade menor
(descarta 0s e 1s que os humanos, em tese, não conseguem ouvir)
- **AAC: Advanced Audio Coding:** geralmente "dentro" de vídeos, Mac
- **WMA: Windows Media Audio:** em computadores Windows
- ...

Serviços de streaming: não transferem arquivos para você, enviam um fluxo (stream) de 0s e 1s que são "tocados" pelo seu computador.

Como representar música? analógico para digital



Edward Ball, na Wikipedia

(https://en.wikipedia.org/wiki/File:Middle_C,_or_262_hertz,_on_a_virtual_oscilloscope.png)

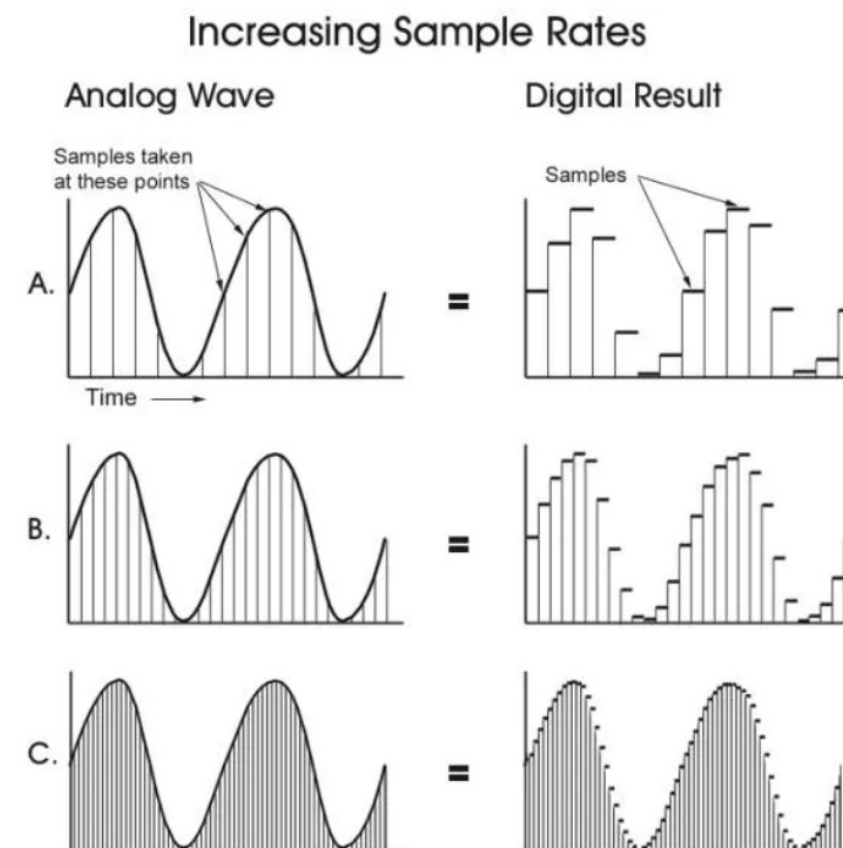
Ao gravarmos uma música, cada som (tom) é uma onda analógica sinusoidal que é caracterizada por:

- **Duração:** por quanto tempo a nota é ouvida
- **Frequência:** sons mais graves ou mais agudos
- **Intensidade:** a altura do som, mais alto ou mais baixo
- **Timbre:** permite distinguir sons do mesmo tom ou nota que foram produzidos por fontes sonoras diferentes (piano e violino)

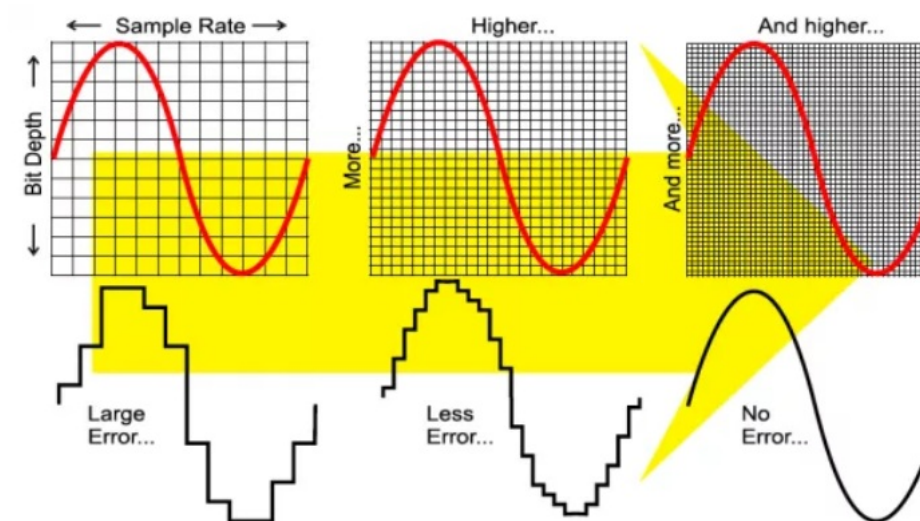
Qualidade da música gravada: analógico para digital

Pelo menos 2 parâmetros:

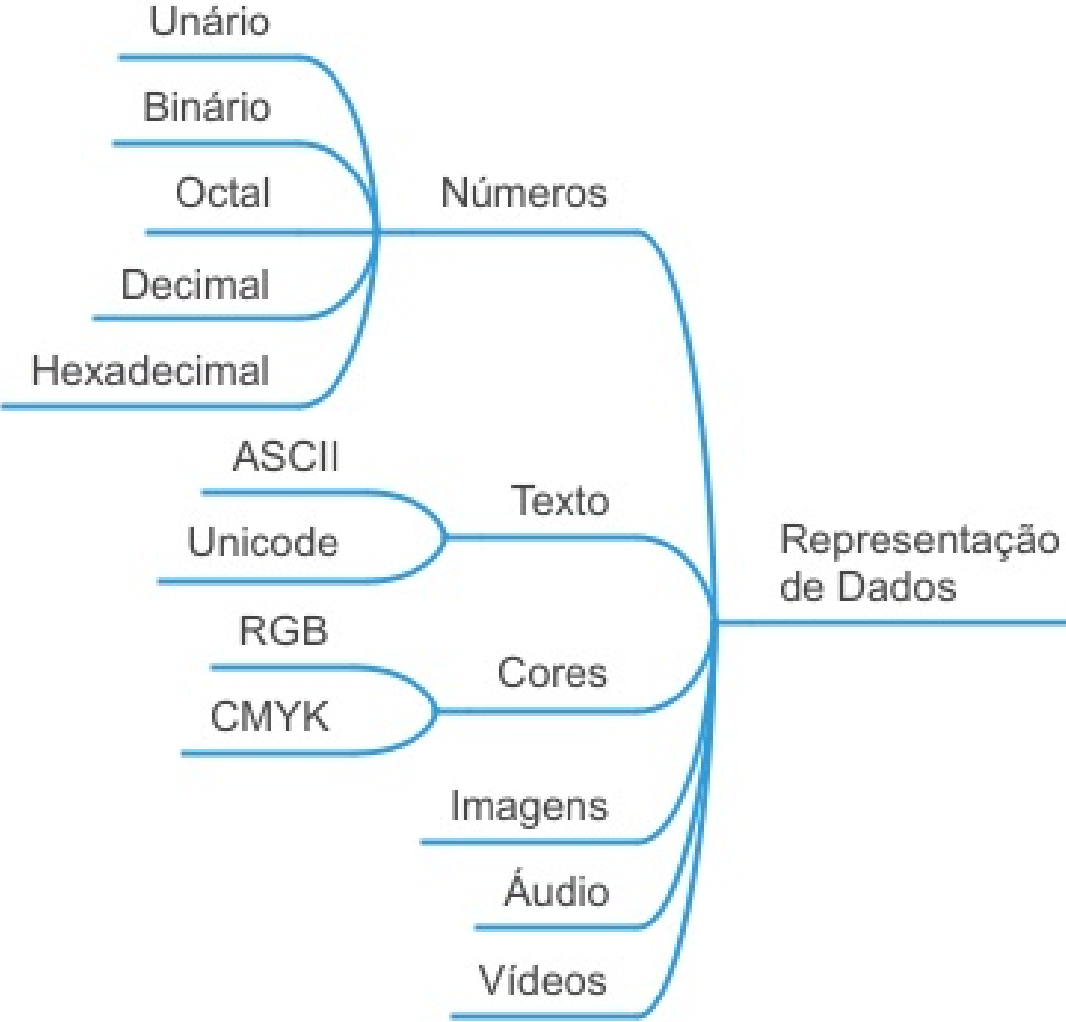
- **Taxa de amostragem** (sampling frequency):
O número de vezes, por segundo, que o som é gravado (44.1 kHz, **48 kHz**, 96 kHz, 192 kHz)
- **Profundidade de bits** (bit depth):
Quantos bits são utilizados para registrar o som, em cada amostra (16, **24**, 32).



O tamanho do arquivo final será dado por:
amostragem x profundidade x tempo



Representação de dados: vídeos



Arquivo de vídeo

Compressão de vídeo

Formatos de vídeo

O que é um arquivo de vídeo?

Basicamente: várias fotos/imagens em seqüência, exibidas rapidamente, dando a impressão de continuidade.

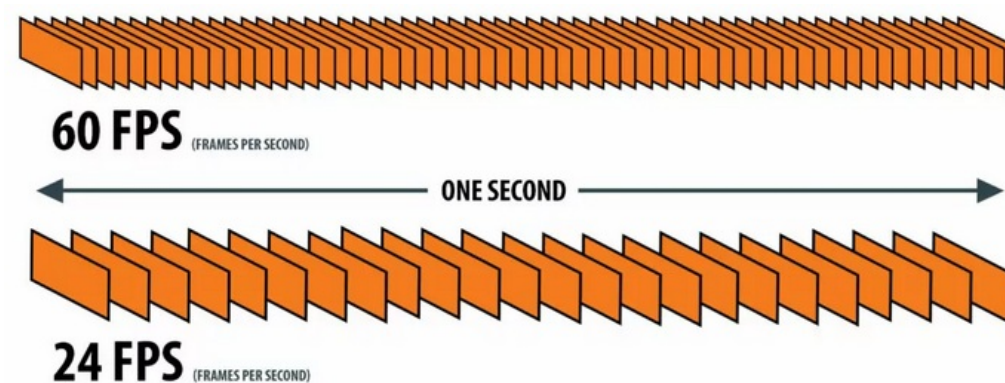


Andymation, no YouTube
(https://www.youtube.com/watch?v=p3q9MM__h-M)

Cinema: 24 frames/s (FPS)

You Tube: 30 ou 60 FPS

Games: 60, 120+ FPS

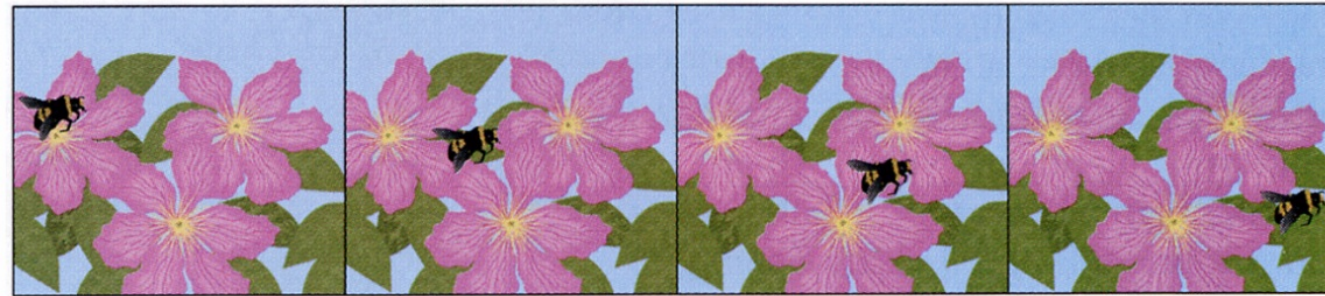
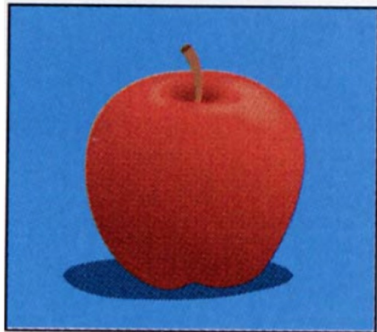


eMania News
(<https://blog.emania.com.br/videos-em-24fps-30fps-e-60fps-qual-taxa-de-quadros-usar/>)

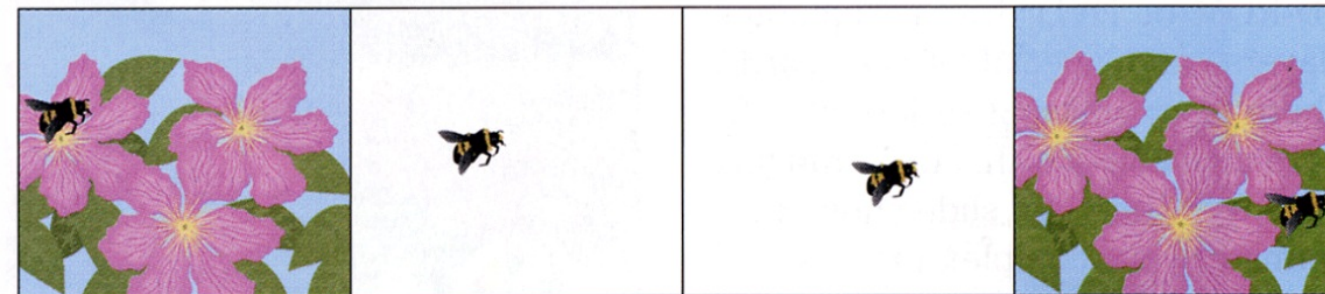
Compressão de vídeos

Não é um armazenamento simples de imagens, há muitos algoritmos matemáticos para compressão:

- **intra-frame**: como nas imagens
- **inter-frame**: armazena as diferenças entre frames adjacentes mantém "key frames"



Uncompressed video



Compressed video

Formatos de arquivos de vídeo

Um pouco mais complicado: imagem, som e outras coisas.

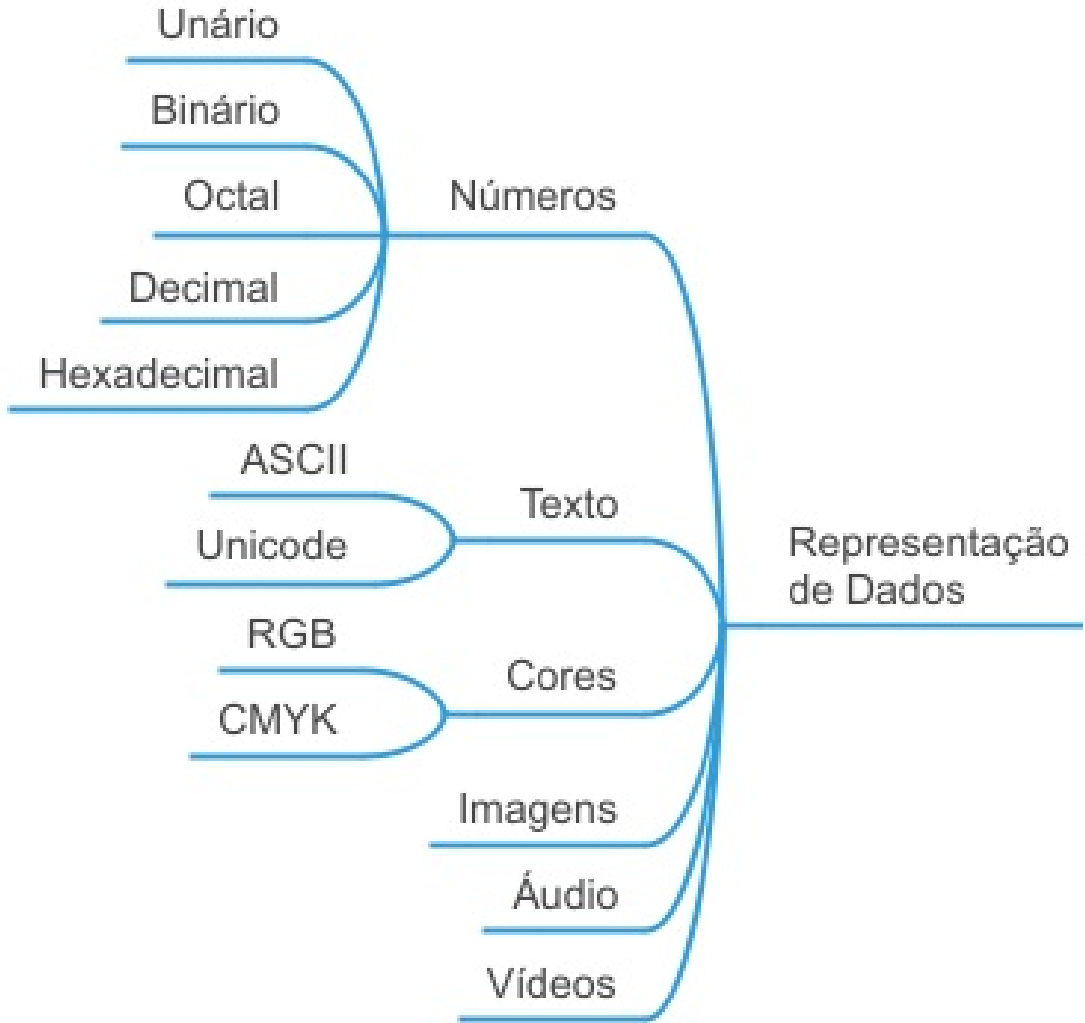
Containers:

- são "caixas" digitais nas quais são colocadas múltiplos tipos de dados
- podem incluir trilhas de áudio, vídeo, legendas, etc...
- principais containers:
 - AVI e DivX: populares no Windows
 - MP4: universal, browsers
 - Matroska: open source
 - QuickTime: MacOS

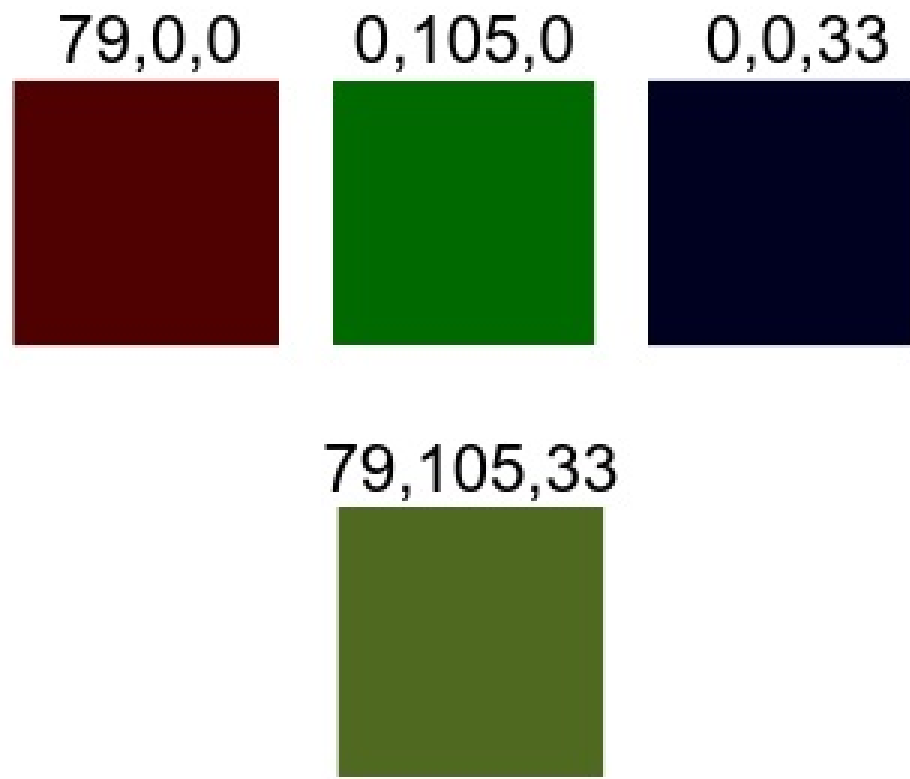
Codecs:

- diferentes modos de fazer o encoding dos dados
- para vídeo: H.264, MPEG-4, ...
- para áudio: AAC, MP3, ...

Representação de dados: tudo é binário!



Se tudo é binário, como diferenciar as coisas?



Oi!

79 105 33

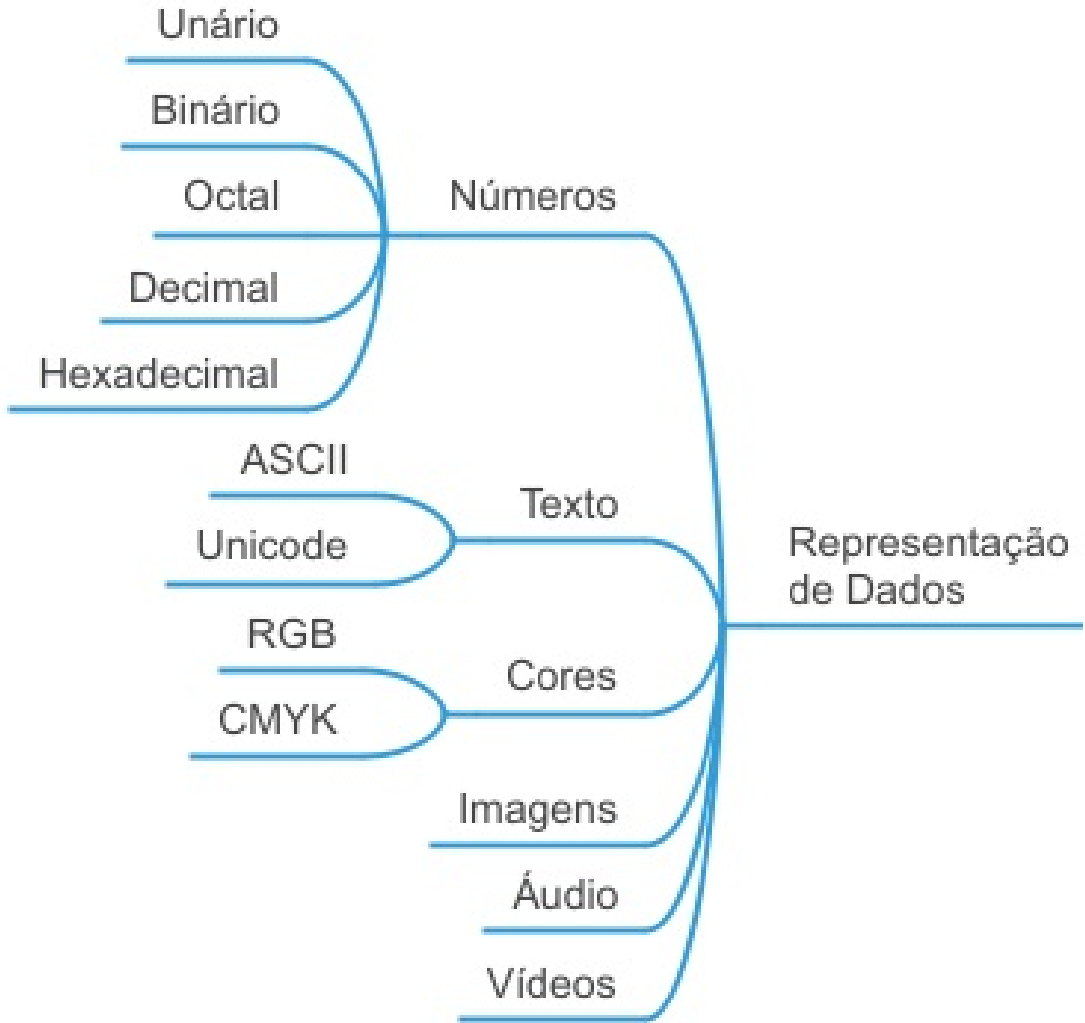
1001111 1101001 100001

010011110110100100100001

010011110110100100100001

Pelo **contexto**! Todo padrão de bits tem múltiplas interpretações, não existe um modo fixo determinado para interpretar nenhum padrão de bits. Os softwares entendem o contexto dos dados e interpretam o padrão binário da forma correta: **a interpretação correta é determinada apenas pelo uso dos bits de um modo específico**. Isso é assim na linguagem natural: "Maria comeu a **manga**".

Resumo: 2º grande fundamento da computação: representação de dados



No próximo vídeo: o coração da computação, algoritmos!

